THE MAGAZINE FOR COMPUTING & DIGITAL MAKING EDUCATORS

(Hello World)

Issue 4 | Spring Term 2018 helloworld.cc

THE ROYAL SOCIETY: AFTER THE REBOOT

Learn about the latest report, which finds numerous challenges still exist in computing education

PROFESSIONAL DEVELOPMENT

Join our panel of experts as they reveal successful approaches to CPD

THE CYBER GAMES

New programme is looking for the next generation of security experts

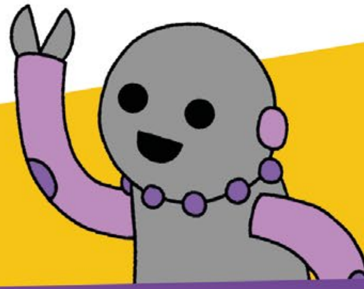
HOW TO BECOME A COMPUTING EDUCATOR

Expert advice on transitioning

RUNNING PUBLIC EVENTS

Health and safety, money, safeguarding, and more

PLUS OBJECT ORIENTED CODING WITH PYTHON • ENGAGING STUDENTS WITH DRONES • CS UNPLUGGED • MICRO:BIT MATHEMATICAL MUSINGS • DIGITAL LITERACY: LOST IN TRANSLATION? • SCRATCH SPELLING TEST • CPD RESEARCH



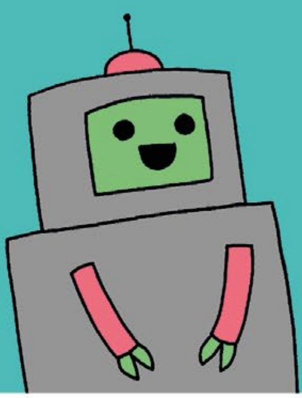
START A CODE CLUB IN YOUR SCHOOL!

Code Club is a network of volunteers and educators who run free coding clubs for young people aged 9-13.

Our aim is to inspire the next generation to get excited about computer science and digital making.

"We use Code Club's fun educational resources to run a weekly after-school club for Year 7 and Year 8 pupils. The students benefit considerably from the extra challenge!"

Karen Dadd, Computing Teacher



- **Code Club is free**
- **Code Club provides step-by-step guides for Scratch, Python, HTML, and Sonic Pi**
- **Code Club helps children develop skills including logical thinking, creativity, and resilience**

We have over 6000 clubs across the UK teaching more than 80,000 young people to code—come and join us!"

Find out more at www.codeclub.org.uk

Code Club is part of the Raspberry Pi Foundation. Registered Charity Number 1129409

(Hello World)

CELEBRATING **15,000** SUBSCRIBERS

HELLO, WORLD!

SUBSCRIBE IN PRINT FOR FREE TURN TO PAGE 28

November 2017 was a pretty exciting month for UK computing education.

On 10 November, we had the long anticipated publication of the Royal Society's report on computing education, After the Reboot, making some key recommendations for building on, and extending, the perhaps fragile successes so far – see Oliver Quinlan's helpful summary of this work on page 6.

Less than a fortnight later, the budget saw Philip Hammond announce £84m of funding to ensure proper, effective CPD for secondary teachers to retrain in Computer Science, so that every secondary school has someone qualified to teach CS at GCSE level – getting this professional development right is going to be a priority. We've some great advice on what computing CPD should be like from David Weston, Simon Humphreys, Gary Stager, and others in our cover feature, starting on page 18.

Also check out stories from Chris Swan, James Robinson, and Charlotte Rubringer about how they got involved in CS teaching, see how their stories compare with yours.

Within a week of the budget, Ofqual announced a consultation on the non-exam assessment (NEA) in GCSE computing, responding to widely reported misconduct. We've since learned that pupils' work on the NEA will no longer count towards their grades. Let's hope that, even without credit for the NEA, practical coding will remain a vital part of the courses leading up to these exams.

There's plenty of other great content here, for example: Greg Michaelson's take on programming paradigms, some brilliant lesson plans to try out, and inspiring examples of a wide range of physical computing projects.

We'd love to hear from you! If you've an idea for a story, want to share a lesson plan, or have computing or digital making questions to answer, please do ping us an email – contact@helloworld.cc.

Miles Berry
Contributing Editor

FEATURED THIS ISSUE

SIMON HUMPHREYS
CAS NATIONAL COORDINATOR

Simon is the National Coordinator for the Computing At School Working Group, which roughly translates to steering the direction and ethos of CAS, and managing the day-to-day activities of the CAS group.

JANE WAITE
CAS LONDON PROJECT MANAGER

Jane is a former industry developer and primary school teacher. Jane is now a Project Manager with CAS London and works in research at Queen Mary University of London, and King's College London.

GARY STAGER
UNIVERSITY PROFESSOR

Veteran teacher, educator, and speaker. Gary curates the Papert archive at **dailypapert.com,** and is co-author of the highly recommended Invent To Learn - Making, Tinkering, and Engineering in the Classroom.

(HW)

EDITORIAL

Publishing Director
Russell Barnes
russell@helloworld.cc

Contributing Editor
Miles Berry
miles@helloworld.cc

Sub Editors
Nicola King and Jem Roberts

DESIGN

Critical Media
criticalmedia.co.uk

Head of Design
Dougal Matthews

Designers
Mike Kay and Lee Allen

Illustrator:
Sam Alder

Cover photography:
Brian O'Halloran/Raspberry Pi Foundation

CONTRIBUTORS

Lucy Hattersley, Rob Zwetsloot, John Stout, and Rachel Churcher

Sponsored by BT

Hello World is a joint collaboration:

Raspberry Pi
bcs Academy of Computing
COMPUTING AT SCHOOL EDUCATE · ENGAGE · ENCOURAGE

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.

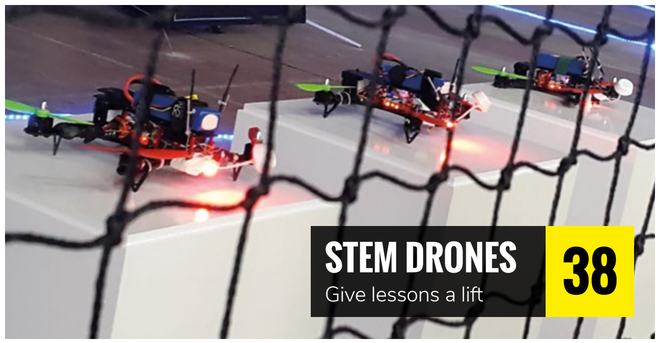
Hello World is published by Raspberry Pi (Trading) Ltd., 30 Station Road, Cambridge, CB1 2JH. The publisher, editor, and contributors accept no responsibility in respect of any omissions or errors relating to skills, products or services referred to in the magazine.

(HW) CONTENTS

COVER FEATURE 18

PROFESSIONAL DEVELOPMENT
Experts offer their successful approaches

CYBER GAMES 11
A programme to find the next generation of cybersecurity experts



STEM DRONES 38
Give lessons a lift

NEWS AND FEATURES

CONVERSATION

REVIEWS

LEARNING

TUTORIALS & LESSON PLANS

Image © The Royal Society

ROYAL SOCIETY REPORT

The 'After the reboot' report finds challenges still exist in computing education

STORY BY Oliver Quinlan

WHO ARE THE ROYAL SOCIETY?

The Royal Society is the UK's national science academy, a fellowship of scientists, and the oldest scientific academy still in existence. Its mission is to promote excellence in science, support international collaboration, and demonstrate the importance of science to everyone. It does this through programmes, research, and policy work. Since the 1660s, fellows of the Royal Society have been involved in some of the most important scientific discoveries made. The Royal Society is an independent organisation, working independently of government, and structured as a charity.

Five years on from its influential 'Shut down or restart?' report, the Royal Society has taken another look at the state of computing education in the UK. It has found that, while there has been some progress, computing education in the UK is 'patchy and fragile'. Although many educators have been working hard to provide the best opportunities for young people, they have not been getting the support they need, and a range of challenges have prevented computing opportunities becoming available and attractive for all young people.

In the last five years, much has changed for computing education in the UK. Programming and computer science are now being taught across the four UK nations and, in England, the ICT curriculum has been entirely replaced with computing. Government policies may have changed, but what has really changed in schools and classrooms? In order to find out, the Royal Society reached out to over 900 schools across the UK, asking them about their levels of confidence, resources used, and how they have implemented the curriculum. They also met with teachers across the country to dig into the details of the survey findings. This was supported by literature reviews on the evidence of effective computing teaching and assessment, and a review of the datasets on examinations, the school workforce, and universities.

Things to celebrate

In the last few years, computing has become an entitlement for state schools across the nations in the UK. Each of the four nations is responsible for its own

Image © The Royal Society

education systems and curricula, and they all now set out the opportunities for young people to learn computing. The subject is defined by the Royal Society as including the traditional IT skills, but also the areas of digital literacy and the more technical area of computer science. This is a significant change from where we were in 2012, where the core ICT lessons available to most students had little in terms of understanding of how technology works and how to create with it. Educators across the UK have worked hard to make these significant changes in their teaching in a relatively short space of time, and celebration of that is due.

Extra-curricular opportunities for young people to get involved in computing are called out as being a particular success. 62% of primary schools and 72% of secondary schools provide these opportunities for their students. The Duke of York award, Code Club, and CoderDojo are examples given of extra-curricular activities that are enriching the experience of computing across many schools.

Problems

Despite this great work, the report describes some significant problems with computing in the UK, some of which will be recognised by teachers. Teacher supply and confidence, lack of training, the gender gap, and the lack of opportunities for young people to study for exams are all highlighted as holding back the subject. There is more to be done on qualifications, and the lack of research and evidence around teaching this new subject to children and young people is also raised. To address these, the Royal Society provides twelve recommendations it thinks need to be put in place to support teachers.

> “ THIS REPORT STRONGLY ACKNOWLEDGES THE CHALLENGES THAT HAVE COME WITH THE INTRODUCTION OF THE NEW SUBJECT

SHUT DOWN OR RESTART?

'Shut down or restart?' was the last comprehensive review of computing education in the UK carried out by the Royal Society. It was published in January 2012, and was an important influence on the changes to curricula across the UK that have happened since. It found that there was a lack of computing opportunities for young people across the UK, particularly in computer science. You can read the report at **helloworld.cc/2AbfLdy**.

Teacher supply and confidence

Computing teachers will be glad, although probably not surprised, to see that this report strongly acknowledges the challenges that have come with the introduction of the new subject in schools. 44% of secondary teachers in the survey said they only felt comfortable with the earlier stages of the curriculum, with 32% of primary teachers reporting they felt similarly about their curriculum. The survey found there is huge diversity in the knowledge and experience of computing teachers. Although some teachers are confident subject specialists, it also identifies those who are optimistic but still learning, and those who lack confidence and feel they need significantly more training. Some teachers also perceive that their students know more than they do about technology, although the authors of the report do counter that students often know how to use technology well, but much less about how it works.

The report estimates that less than 30% of secondary computing teachers have a degree in the subject, compared to 51% of physics teachers and 46% of maths teachers. However, 7% of primary teachers who responded had a background in computing, which suggests the survey attracted a higher proportion of those with such backgrounds than is representative of the teaching population as a whole.

The lack of confidence of existing teachers is a problem, but we also need to see more

Image © The Royal Society

new people choosing to become computing teachers to replace those who retire or leave teaching. Finding people to take on these roles appears to be as much of a challenge, and the government in England only reached 68% of the recruitment target for 2012-2017. Prospective computing teachers have skills that are sought after by industry, and it seems currently teaching is not appearing to be as attractive a proposition for them. The incentives that do exist can also have perverse consequences. Currently, newly qualified teachers who took a tax-free bursary to train can see their real earnings drop when starting a job, hardly an incentive to stay. Many drop out well before then, due to the pressures of the training course.

The report's authors recommend that higher education providers do more to promote careers in education to those beyond the 6% of computing graduates that currently go into education. The idea of 'braided careers' is also recommended, to encourage people to mix an industry career with one in teaching.

Lack of training

The answer to challenges of teacher confidence is professional development, but 'After the reboot' finds that, to date, this has not been available to enough teachers to make the difference that is needed. Many serving teachers have the dual needs of developing their subject knowledge in computing as well as the pedagogical approaches to effectively teach it. The report found that teacher network meetings and external courses are happening, but that there are gaps in sustained in-school development such as mentoring, action research, and self study. Then there is the issue of whether teachers are able to access these opportunities; the teacher survey found that 29% of primary and 26% of secondary teachers report experiencing no professional development activities in the 2015-16 academic year. However, good professional development is happening, it just needs to be scaled up. In particular, the Computing At School Network of Excellence is a channel for professional development that is providing many opportunities, but needs to be funded to grow and reach the proportion of teachers who have not been able to engage in vital professional learning. As well as national funding, this will require school leaders to prioritise professional development in this subject as, at the moment, many teachers are undertaking their learning in their own time.

Widening access

Media coverage of computing education often includes the male-dominated nature of the subject in schools and careers. Only 20% of GCSE entries in 2017 were girls, and only 13% of undergraduates studying computer science are women. The report found that more girls think of computer science as difficult, not interesting, and not needed for their career plans. More girls choose to study computing at single-sex schools and, when they do, they often do as well or better than boys in their exams. The authors suggest this shows that the environment can have a lot to do with the choices they make. Their literature review suggests that working on school culture and ensuring that computing is not presented in a gender-biased way could have an impact. However, the main recommendation is that more research is needed into how we can help more girls to see the opportunities computing could present them.

Gender isn't the only area where equality of access to computing is a problem. Pupils who live in areas which mostly have smaller schools are less likely to have an exam course in computing provided for them. This creates a contrast between urban

TIMELINE

2010
The 'Curriculum for Excellence' was introduced in Scotland, with computing science as an area of study

Jan 2012
'Shut down or restart?' published, calling for more computer science in the curriculum

Jan 2012
Education Secretary Michael Gove disapplies ICT programme of study in the national curriculum

Sept 2014
New subject of computing begins being taught in English schools

Sept 2016
Digital Competence Framework launched in Welsh schools

Sept 2016
GCE Digital Technologies qualification launched in Northern Ireland

Nov 2016
Technologies Experiences and Outcomes for Scottish schools refreshed by Education Scotland

areas, where computing qualifications are often available, and rural areas where they often are not. More socially disadvantaged students are also less likely to study computing. There are also trends towards some ethnic groups being less likely to study computing, and teachers have concerns about provisions for students with special educational needs and disabilities. These trends all show that those studying computing do not currently represent the diversity of our society.

Qualifications

The GCSEs in computer science in England, Scotland, and Northern Ireland are judged in the report to have been a success, with more students taking them and more students progressing to study A Level computer science. However, the Royal Society's previous report recommended there should be a range of qualifications available to students between 14 and 16. With the phasing out of GCSE ICT, the study of aspects of computing, other than computer science, at this level are left to a range of qualifications that they are concerned lack credibility with teachers. They provide a recommendation that Ofqual should work to ensure there are qualifications in IT for these age groups. ICT advocates have, for some time, been noting the strong focus on the computer science aspects of the curriculum, and the creation of additional opportunities in IT will be welcome news to them, and likely provide a chance for more young people to gain qualifications in digital technology.

More research needed

While the report shows there are practical gaps in computing education in England, there is also a big gap in our understanding of how people learn in the subject. Most research to date has involved undergraduates. Now that we are teaching similar concepts to children and young people, a lot more work is needed on how young students learn the practical skills and abstract concepts of computing.

Several areas are identified as needing more research. Better understanding of learning models and instructional techniques would allow teachers to design lessons that are most effective for students to learn the content and skills of the subject. The context of learning can vary hugely, and affects both how students learn and how they engage with computing. The authors identify physical computing as an area with promise here, but we don't yet know enough about how it supports student learning. The programming languages used have a similar effect on learning. Now educators are facing questions such as how to transition from visual to block-based programming languages. Student engagement is also identified as an area we need to better understand, not least to make the subject appealing to more diverse groups of students. Finally, assessment techniques allow teachers to understand how students are learning and support them to get to the next steps.

The Royal Society recommends a strategic plan for computing education research in the UK, including a long-term agenda for research and a commitment to effective sharing of research with teachers.

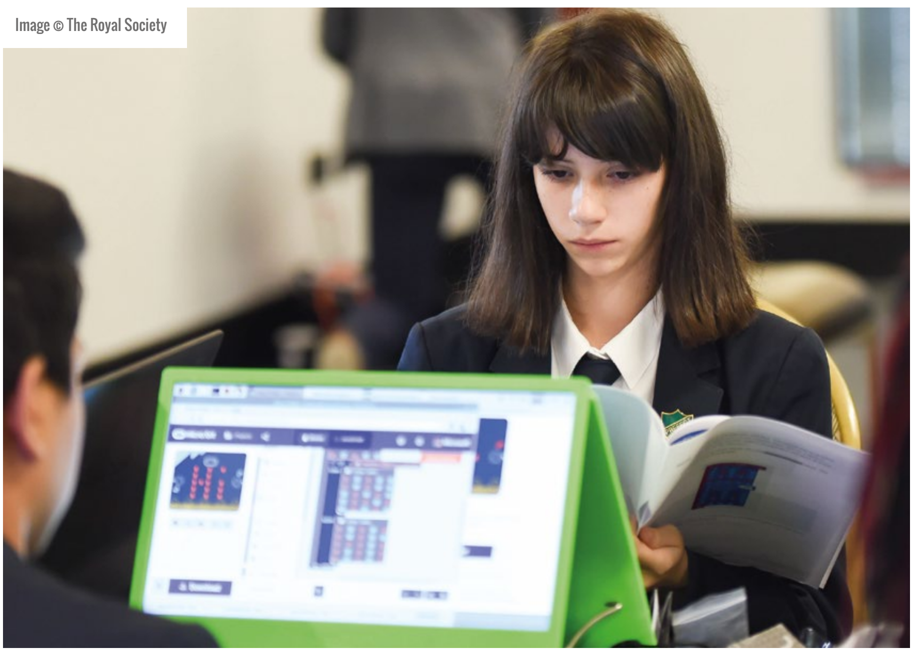


Image © The Royal Society

DIGITAL JOBS

The report found that employers are projecting a big growth in jobs depending on computing skills. Cybersecurity is one example, currently making up 15% of UK IT jobs and set to grow 10% each year until 2020. Machine learning is another emerging area which needs lots more well-prepared young people for the growing jobs. As well as specialist IT jobs, the report shows that computing skills will become ever more important in a range of other jobs, from project managers developing digital solutions, to scientists using simulations instead of traditional experiments.

The future for UK computing education

The Royal Society's 'Shut down or restart?' report in 2012 set a challenge for UK education of a scale that is still becoming clear. 'After the reboot' finds much to celebrate, but some huge challenges stand in the way of the vision of a new computing curriculum for all children in the UK. Perhaps the most immediate to teachers are the difficulties related to professional development. In its autumn budget, the UK government has just announced £100 million of new funding for training computing teachers. This addresses one of the central challenges of the report, but there are still others relating to qualifications provision, diversity, research, and the recruitment of new teachers. The report paints a picture of a subject that needs urgent attention to build on an enthusiastic but 'patchy' start. You can read the full report at **helloworld.cc/2jn52px**. (HW)

LIFE IN LIKES

New report on the effects of social media on 8-to-12-year-olds

The way children use social media, and its effects on their wellbeing hits a "cliff edge" when they start secondary school, says a new report from the Children's Commissioner.

'Life In Likes' is a research report on how younger social children use social media platforms, which aren't designed for their use.

Whilst most social media sites have an official age limit of 13 years, the report suggests that up to three-quarters of 10-to-12-year-olds have a social media account.

Social changes

According to the report, 8-10-year-olds use social media very differently from 11-12-year-old children. Younger children use social media in a playful, creative way (typically to play games.). This changes when children's social circles expand in year 7.

At this point, many children often find social media difficult to manage and it's all too easy to become dependent on likes and comments for social validation. Many children also adapt their offline behaviour to fit an online image.

Anne Longfield, the Children's Commissioner for England, launching today's report, said:

"While social media clearly provides some great benefits to children, it is also exposing them to significant risks emotionally, particularly as they approach Year 7."

"It is also clear that social media companies are still not doing enough to stop under-13s using their platforms in the first place."

"I want to see children living healthy digital lives. That means parents engaging more with what their children are doing online. Just because a child has learnt the safety messages at primary school does not mean they are prepared for all the challenges that social media will present. It means a bigger role for schools in making sure children are prepared for the emotional demands of social media. And it means social media companies need to take more responsibility."

Longfield calls on schools and parents to prepare children for this change towards the end of primary school. She also calls for compulsory digital literacy and online resilience lessons for Year 6 and 7s, so that they learn about the emotional side of living with social media, and not just messages about safety.

You can read the full report on the Children's Commissioner website: **helloworld.cc/2D792UY**. (HW)

The Children's Commissioner's report on the effects of social media on 8-to-12-year-olds examines the way children use social media

Students at UCL Academy are amongst the first to try out stage on of the programme: CyberStart Assess

CYBER GAMES

New Cyber security programme uses gamified learning to find the next generation of cyber security experts

Cyber Discovery is a new cyber security programme launched by The Department for Digital, Culture, Media and Sport (DCMS).

We first reported back in Hello World #2 that the DCMS had announced £20m to fund cyber security to UK students. The Cyber Discovery programme is now available for students in years 10-13.

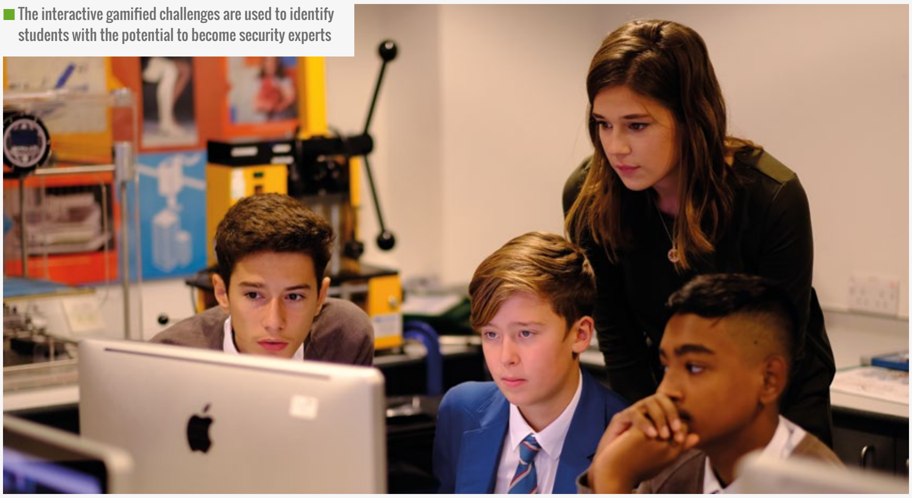
"Cyber Discovery will help inspire the digital talent of tomorrow and give thousands of young people the opportunity to develop cutting-edge cyber security skills and fast-track future careers," said Karen Bradley, the Secretary of State for Digital, Culture, Media and Sport.

Gamified testing

The course is being delivered by a consortium of relevant organisations: SANS Institute, BT, Cyber Security Challenge UK, and FutureLearn.

Cyber Discovery uses gamified learning to nurture and develop crucial cyber security skills in students. The curriculum will cover everything from digital forensics, defending against web attacks and cryptography, to Linux, programming and ethics. Cyber Discovery consists of four stages. Initially, students are invited to register and work through a selection tool, CyberStart Assess. Successful students will go onto three challenging and exciting stages which will later include Face to Face camps with industry experts, and three live regional Capture the Flag events, where parents and leaders can see the progress made by students.

The interactive gamified challenges are used to identify students with the potential to become security experts



Debbie Tunstall, Head of Education at Cyber Security Challenge UK, said: "Cyber security is an industry that's still in its infancy, meaning very few young people know and understand that there are lucrative careers awaiting them in the field. With a critical skills gap looming and the cybercrime threat growing, we need to educate about cyber security while individuals are still young; peaking their interest in future cyber careers and, as a result, filling the pipeline of talent. The Challenge has years of experience in dealing with people in this age group and providing fun and educational face-to-face events and we're delighted to bring our expertise to this innovative programme"

Cyber Discovery is being piloted in year 1 in England, but is expected to expand to other parts of the UK in years 2, 3, and 4.

Potential students and club leaders can sign up at **joincyberdiscovery.com**. (HW)

The five new Pi Cafés will operate much like the pilot site in Broadoak School, Partington

PI TECH CAFÉS

Five Raspberry Pi cafés for Manchester Schools

Following a successful pilot in 2015, web-hosting firm UKFast has announced five Raspberry Pi Cafés for Manchester schools to be set up in 2018. The project represents a £100,000 investment from UKFast.

Aaron Saxon, UKFast's Director of Training and Education, said: "We are distributing 120 Pis across the five sites: Holy Name RC Primary School in Moss Side, St Bedes Prep School in Hulme, Alderley Edge School for Girls, The Hollins Tech College in Accrington, and Factory Youth Zone in North Manchester."

The sites were chosen "where gaps in digital engagement exist", Aaron explained. This includes areas lacking "the resources to deliver cutting-edge digital training, as well as all-girls schools which have traditionally seen a low uptake in technical subjects."

> WE'RE PROVIDING TECHNOLOGY FOR THE CHILDREN IN A FUN AND EXCITING WAY

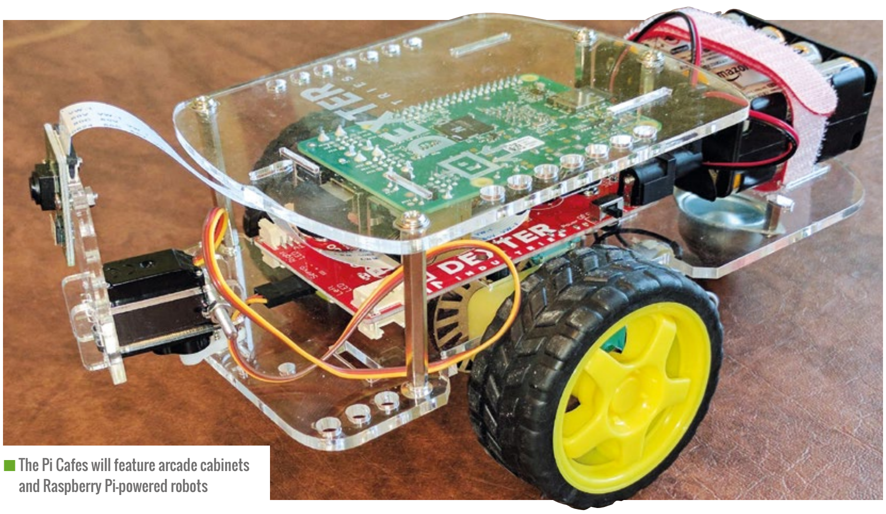
Creative space

"Some schools may use it as a creative space, others will use it as their computer science classroom, as well as an extracurricular hub and space for the community," said Saxon

The Raspberry Pi computers "won't look like traditional desktop units," Saxon told us, "as we want them to be more computer-science focused." For UKFast, that means "there will be arcade, old-school gaming and robotics cases" Saxon revealed. "We're providing the technology for the children in a fun and exciting way."

Paul Grier, Network Manager at St Bedes Prep School (one of the five new sites), added that "in 20 years' time, 45% of jobs will be done by AI and robots. So, if kids today don't understand [these things], they won't understand how the world works."

Grier added that he hopes the new Pi Café will "allow both children and the staff [of St Bedes] to delve more into computer science." While students and staff of St Bedes "learn ICT, which is processing and spreadsheets," Paul explained that, "programming hasn't taken off as much as I would have liked it to." (HW)



The Pi Cafes will feature arcade cabinets and Raspberry Pi-powered robots

BIXELS: THE BIOCOMPUTING DISPLAY

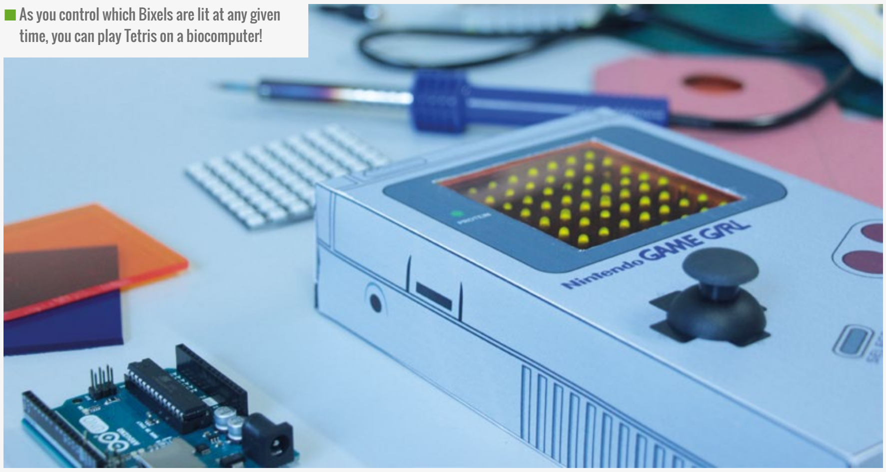
Irish firm using DNA screen to teach kids bio-computing

Irish bio-tech firm Cell-Free Technology has launched a Kickstarter campaign for a “world-first demonstration of a DNA programmed bio-computer that can play Tetris”.

Bixels is an 8 × 8 grid of ‘bio-pixels’ that can be controlled from a smartphone. As you can electronically control which Bixels are lit, the Bixels act just like the pixels on your screen.

The DNA is synthetically replicated from the same DNA that allows a jellyfish to glow green – no jellyfish are harmed to make Bixels.

Cell-Free Technology CEO Dr Thomas Meany tells us, “Bixels is a hugely valuable resource for anyone who needs a low-cost way to study fluorescent proteins in a lab, but our real target is STEAM educators.” As Meany points out, Bixels “incorporates almost every aspect of a STEAM curriculum in a single workshop.”



As you control which Bixels are lit at any given time, you can play Tetris on a biocomputer!

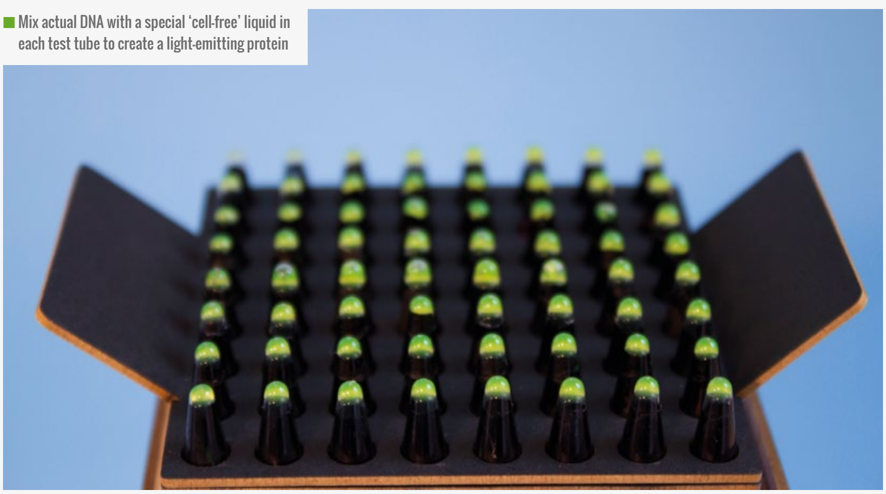
DNA for the masses

Bixels places an 8 × 8 grid of small test tubes (called PCR tubes) onto an 8 × 8 grid of RGB NeoPixels (controlled by an Adafruit Bluefruit Feather). By mixing the liquid in the two coloured vials of the kit in each PCR tube, you create a mixture that emits green when the NeoPixel beneath shines blue.

Or, as Meany explains: “The blue vial contains cell-free extract which has the nano-machinery (ribosomes, RNA polymerase, and transcription factors) that, when the DNA is added, can be programmed to produce a protein (in our case fluorescent protein)”. A coloured filter within the Bixels housing ensures only the light emitted by the protein is seen.

Bixels is safe to play with and use because of the unique ‘cell-free’ technology developed by Cell-Free Technology. The ‘bacteriophage infection’ used breaks down cell walls without harming the contents, leaving you with a liquid that can be biologically programmed “without the fear of a bacteria or other organism [forming],” confirms Meany.

Bixels has a target of €9267 (£8166), with a basic Bixel Solo kit only costing €90 (£79) See **helloworld.cc/2D71ilW**. (HW)



Mix actual DNA with a special ‘cell-free’ liquid in each test tube to create a light-emitting protein

#INSIGHTS

PROFESSIONAL DEVELOPMENT: WHAT THE RESEARCH SAYS

STORY BY Oliver Quinlan

Professional development can be hugely valuable, but educators are often limited in the time available for it and are faced with a huge range of options that can be difficult to choose from. With all this choice, what will have the biggest impact on the young people you work with?

In 2014 The Teacher Development Trust assembled a team of education professors and researchers to look at the evidence for what makes the most effective professional development for teachers. Using international research, they distilled eight principles of the most effective courses for teachers.

Duration and rhythm

Many courses for teachers are only a single day. It can be helpful to get out of school and focus, but the research suggested that it is the programmes that last longer than that which have a real impact. Learning about a new approach is just the start, you need to have time to keep revisiting it, applying it in practice, and reflecting on the results. Many people aim to do this with material from a one-day course, but the best professional development gives teachers a structure to support them to frequently check in over up to a year, and explore how the new learning relates to their own experience and practice. Look out for professional development that takes place over a longer term than just an isolated day.

Considering participants needs

We know it's important to understand students' needs and different starting points for learning, and it's no different for teachers. The best professional development has deliberate opportunities for teachers to involve their day-to-day experiences, and set goals based on the needs of their own contexts. Sharing needs with other participants can also allow a shared sense of purpose to develop, which helps in making these goals a reality back in the classroom.

Alignment of content and activities

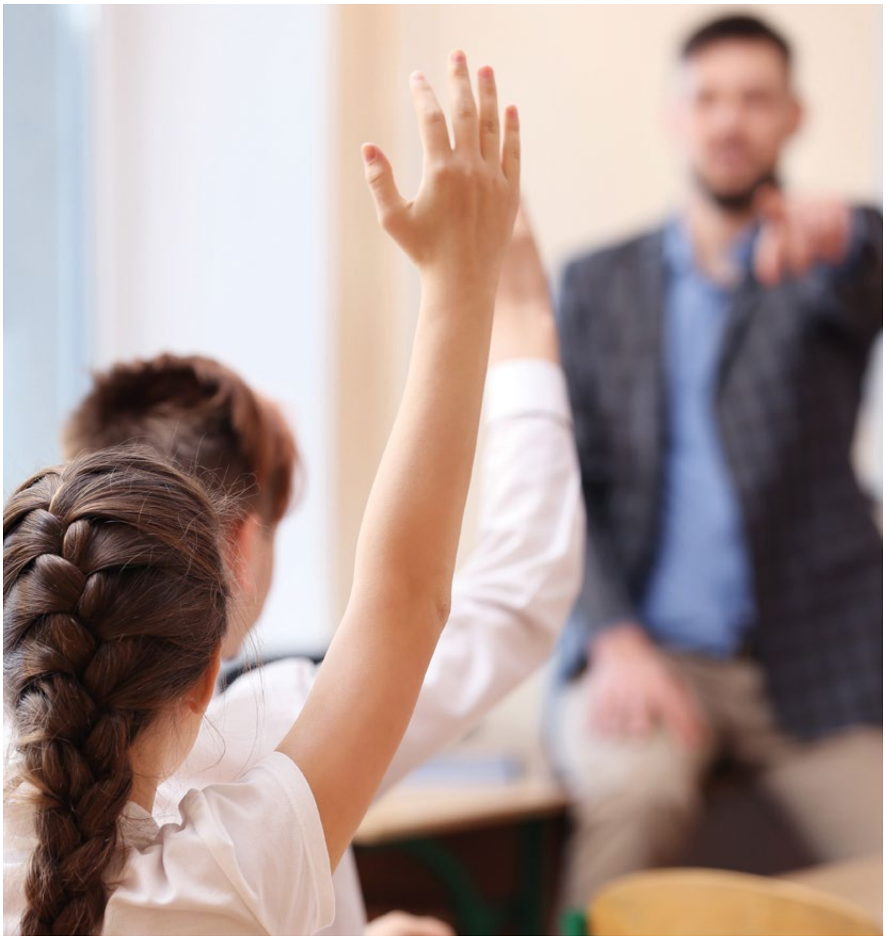
If you're learning about project-based skills, does it make sense to do this through a lecture? The most effective professional development is delivered in a way that aligns with the approach being developed. Look out for, not just what courses are about, but how they are delivered.

Content

The evidence suggested that the most effective professional development tends to focus on certain key building blocks of content. Subject knowledge is the first, with the most effective courses building understanding of the content teachers teach. Linked to this, is subject specific pedagogy; not just how to teach generally, but the detail of how to teach a particular subject. Much of this is about being specific; look out for courses to build subject knowledge and teaching skills in a specific area, rather than to generally improve your generic pedagogy.

Activities

Teachers need time to assimilate new ideas and decide how to fit them into their existing practice. The most effective professional development has activities which allow teachers to consider their context and usual practice, and plan for how they will develop it. Look out for activities that will explore how to translate what you learn into your existing practice. It's important to have a chance to be reflective and use assessment to see what difference new approaches are making to students' progress. Longer-term programmes give a chance to do this in a structured way, which contributes to the findings that they are more effective than one-off day courses.



Focus on how the young people you teach will ultimately benefit

External providers and specialists

The research found external trainers can have an important role. They bring a fresh perspective, and notice and challenge established ways of doing things in a school. However, it's important that teachers are active participants and take the lead in professional development. Just as students need to actively engage with what they learn, teachers do as well. Professional development that has you doing, practicing, planning, and making changes is likely to be the best use of your time.

> IT'S IMPORTANT THAT TEACHERS ARE ACTIVE PARTICIPANTS AND TAKE THE LEAD IN PROFESSIONAL DEVELOPMENT

Collaboration

Although working with young people much of the day, teachers often work alone. Collaborating with others is an important feature of the best professional development. Working through problems, discussing how to help students make the most progress, and embedding new approaches across a school as a team can lead to the most sustainable change. Planning together can also bring new ideas to your work and challenge some of the habits you have which could be more effective.

School leaders

Just as collaboration between teachers is important, collaboration with school leaders can increase the impact of professional development. For school leaders, getting involved with the process promotes a learning culture, and helps to focus on the wider impact that you want to see across the school. School leaders are also often needed to navigate potential barriers and make sure resources and support are in place for change to happen. If they are involved, then improvements are more likely to be facilitated to grow.

There's a great deal to look out for here, research shows that professional development can be quite variable, but there are key features that can make it particularly effective. The fundamental one is a focus on the students. That is, after all, what professional development is about; getting even better at supporting students to progress and develop. If you look for one thing when choosing professional development, make it a focus on what the young people you teach will gain.

You can read the full report, 'Developing Great Teaching', from the Teacher Development Trust here: **http://tdtrust.org/about/dgt**. (HW)

EMOTIONS IN EDUCATION: SUCCESS OF CODERDOJOS

STORY BY Lucia Flóriánová

Learning is always, at its best, an emotional experience. Complex tasks can easily result in feelings of frustration, anger, and helplessness that are highly demotivating. A sense of pride, admiration or excitement makes learning seem like fun. This triggers students' interest and motivates them to improve. Emotions associated with subjects significantly influence whether children are attracted to them – a phenomenon which psychologists call 'valence'. This is especially true for computing, which is often perceived as difficult.

Studying the environment of 4 CoderDojos (technology clubs for 7-17 year olds), McKelvey and Cowan identified the main aspects that made computing and digital making enjoyable for learners:

Social environment

Children appreciated that they could talk to each other and meet new people. It created a positive atmosphere and prevented the feeling of isolation at times of failure. A mix of age groups allowed pupils to support each other.

Informal setting

In more successful CoderDojos, the educator was not a pupils' teacher. It also helped if they were taking place out of school hours or context. This made children feel like they were having fun, rather than attending just another lesson.

Lack of limits

Children enjoyed choosing their own way of solving a problem. It allowed them to take control over their own learning, explore options and experiment, which made tasks exciting. Choosing between different ways also shifted the focus of learning from solution to process. Such 'learning as inquiry' deepens children's understanding.

Support, not control

Similarly, successful educators can show the direction to pupils without commanding them. Children's motivation seemed higher when they could own their education and co-create it.

Interesting tasks

When pupils chose a task that interested them personally, they were more likely to work on it during their free time and to be resilient in solving it. Personal interest also increased young people's pride in their finished work.

Creativity and imagination

Young people were most intrigued by the level of creativity involved in CoderDojos. They felt they could create anything they could imagine. This transformed computing into a skill that students longed to enhance.

Aspects of learning that make CoderDojos enjoyable for students can be transferred to other educational contexts. Through activities and approaches that potentiate positive emotions, teachers can increase young people's interest in a subject. A positive emotional attitude (positive valence) increases young people's perseverance and motivates them to actively participate in lessons. Understanding how to make education 'fun' is thus not only important for maintaining children's well-being, but also for the effectiveness and success of teaching. You can read McKelvey and Cowan's original article here: **helloworld.cc/2Ezdpbc**. (HW)

Children enjoy choosing their own way of solving a problem

#INSIGHTS

CHECKING IN AFTER PROFESSIONAL DEVELOPMENT

STORY BY Oliver Quinlan

Since 2014 the Raspberry Pi Foundation have been training educators through the Picademy programme, and building a community of Certified Educators to promote computing and digital making in education. In 2016 we started running an annual survey of these educators to find out more about what they are doing in their schools and organisations. When you are working on something new in education, it's always useful to know what other people are doing, so this research has been published online so other educators can see what is happening.

There was a lot of positive feedback, and evidence that educators who had been through Picademy had reached over 42,000 young people. Most of these teachers are secondary level Computing teachers or primary teachers teaching Computing, but there are some Design Technology and Science teachers who are integrating digital making into their subjects.

It's interesting to see what equipment other educators have access to. It seems most educators are using relatively small sets of digital making equipment with their students. Although having more will always be helpful, it's useful to know that you don't necessarily need class sets of equipment to do worthwhile work with your students.

We asked about educators' perceived competence in various digital making topics, and their confidence in teaching them. What was most interesting about this is they appeared to correlate, so if an educator felt they understood a topic, then they would feel confident in teaching it. This has made us think about how we deliver professional development, and ensure a strong focus on subject knowledge development.

One of the most interesting findings was how many educators deliver digital making in lunchtime or after school clubs. Further research we've done suggests many take what they have learned and deliver it in informal contexts first, moving on to integrating it into their lessons later. The focus of teacher professional development can be primarily what happens in the classroom, but we think this shows that practicing in a lower risk setting with enthusiastic students is a great way to embed skills and experiment until you are ready to implement them in more formal lessons.

> WE ASKED ABOUT EDUCATORS' PERCEIVED COMPETENCE IN VARIOUS DIGITAL MAKING TOPICS

If you want to find out more about what Raspberry Pi Certified Educators are doing, you can read the full report on the survey at **rpf.io/research.** We've already run a follow up for 2017, so look out for results from this soon. (HW)

APPROACHES TO PROFESSIONAL DEVELOPMENT

With recommendations for more professional development in the Royal Society's report, and government funding to support this, our panel of experts present some successful approaches

5 KEYS TO DEVELOPING GREAT COMPUTER SCIENCE TEACHERS

Schools are full of expertise in teaching pupils, but how do we apply the science of learning to help develop teachers?

STORY BY David Weston

Teachers spend hours sitting in training sessions, but how much of this sticks? How much feels useful? If we want to help each other learn and develop, we need to apply the same care in designing teachers' learning as we do to designing pupils' learning.

When my group created the new CPD Standard, we reviewed a massive international literature regarding what works. It will come as no surprise that isolated blasts of information do just as little for teachers as they do for our students. The good news is that we can do much better. When we get it right, CPD feels empowering and relevant, increasing our job satisfaction, and helping us to lead our students to ever-greater success.

Here are five big findings.

01 Generic teaching advice isn't enough

It turns out, it's hard to improve teaching with generic tips and tricks. Rather than spend time hearing about how to ask good questions, I might spend time thinking about great questions to ask about something specific in my subject, for example, great questions to ask when teaching if-then-else code in Python.

02 Subject knowledge really matters

Moving from generic to subject-specific gives me a chance to increase my subject knowledge. I can support this by some self-testing using exams, reviewing the scheme of work to audit my confidence, and then either getting support from colleagues or scheduling time to work through online training.

03 Culture really matters

We want classrooms that are full of trust, open discussion, and mutual support. We also need the same in staff rooms. Teachers, like students, react badly to fear of humiliation, so we need to make sure that every team works actively to create a culture where learning is encouraged.

04 Lead by example

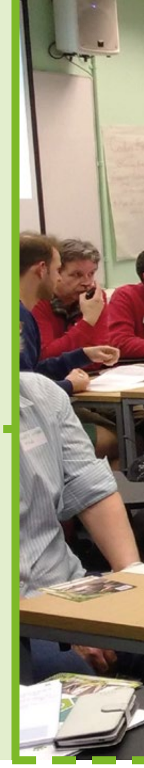
We need school leaders and subject leaders to be the 'lead learners'. Teachers will learn more effectively when senior colleagues talk about their own learning, showing that it's okay to ask for help, to make mistakes, and to share interesting ideas.

05 The best learners are evaluative practitioners

The best CPD starts with two questions: what difference do I need/want to make to my pupils, and how will I know if I've made it? Thinking about impact and evaluation needs to happen before, during, and after teacher development, and it can turbo-charge the learning process. We need all teachers to start with those two questions, and keep pursuing the answers throughout the learning process. As John Hattie likes to say: "teachers, know thy impact".

FOR MORE INFORMATION

This is just the tip of the teacher learning ice-berg. For more information have a look at the DfE CPD Standards (**helloworld.cc/2EZdbeh**), download the TDT/TES Developing Great Teaching review of the research (**helloworld.cc/2ClEw6f**), and sign up for our monthly newsletter of the latest research, news and ideas for great teacher development (**helloworld.cc/2ClcDfr**).



PROFESSIONAL DEVELOPMENT FOR ALL?

How can teachers access appropriate training to support the delivery of Computing in their schools?

STORY BY Simon Humphreys

Over three years into the change from ICT to Computing in our schools, and we are seeing continued demand for Continuing Professional Development (CPD), training, and other forms of support for teachers, but there are many constraints on schools, making it difficult for some to access appropriate training. Computing At School (CAS) has been leading on professional development for teachers through the DfE-funded Network of Excellence and other CAS Community activities. CPD can mean different things to different people but what is clear is that all teachers are engaged in some form of CPD most of the time, from the reflection that takes place after each lesson and notes made about the next time this is delivered, to the conversations with other staff members, to picking up hints and tips from Twitter or Facebook groups to, if we're really lucky, getting permission to attend a day out of school for reflection and training.

At the heart of CAS is our belief that CPD is a human process, not mechanical, something that is not done to teachers but worked on collaboratively with teachers over a period of time.

Ideally, such training is provided locally, face-to-face in a human manner and built from the experience of other teachers, preferably led by them. The notion of building a Personal Learning Network has become very important for teaching, and CAS provides several routes into helping you build your own CPD community, both online and where you work.

HOW DO I FIND MY NEAREST CAS HUB OR CAS MASTER TEACHER?

Visit the CAS page at **computingatschool.org.uk** and enter your postcode. You will then be able to see which CAS Hubs and CAS Master Teachers are closest to you. Clicking on the name of the Hub will then take you to the Hub webpage where you will find contact details. You can subscribe to one or more hub via your CAS profile page.

Attend local hub meeting

There are now over 240 CAS hubs catering for both primary and secondary school teachers throughout the UK, with more in the pipeline. The Hubs are run by teachers, for teachers, and provide the opportunity to meet in a relaxed and informal atmosphere, to share ideas and resources, to receive informal training, and gain mutual support from discussing teaching methods with colleagues. CAS Hubs meet once or twice a term for a couple of hours after school. The style and format of the meeting is up to the discretion of the Hub Leader, but each will focus on an aspect of teaching the Computing curriculum, and will provide plenty of opportunity for teachers to collaborate and share their expertise with one another. You can find your nearest hub via the CAS website (see panel), but if there isn't a hub running in your area why not start one? They're low maintenance but high impact events, and hub leaders can tailor the meeting around their own needs and CAS provides all that you need to get started.

Discuss with other teachers on CAS Online

Each of the CAS hubs is provided an online discussion space through the CAS Community website, but there are other forums available for teachers to post questions and get answers. The CAS Community forum (CAS Online) has over 28,000 registered members. We strive to make it a safe place for all to ask whatever they need to ask (NB. students and pupils are not given permission join the forum) and in the community

Cas London networking meeting

there will also be people with additional expertise i.e. academics, researchers, industry, exam board personnel, governors etc.. Somebody, somewhere in the CAS community, will almost certainly have the answer to your question, you just need to do your research and ask. If you'd rather contact someone privately, then we have recently set up a private messaging system so you can ask individual CAS members questions with confidence.

Attend sessions run by a local CAS Master Teacher

The CAS Master Teachers have been an integral element in the DfE-funded Network of Excellence. The NoE has built on the grass roots ethos, central to Computing At School, through inspiring, leading, training, and supporting an active community of practice delivered by CAS Master Teachers providing local, face-to-face, peer-to-peer CPD.

The CAS Master Teachers are available through the local CAS hub, CAS Online, or email and will be providing CPD sessions in their local area. Get in touch with them if you need specific help by using the same postcode search tool on the CAS website.

Attend a session run by their local university

CAS has good relationships with most of the universities offering Computer Science to their undergraduates. The NoE has established ten regional centres based in universities, to support local delivery of CPD. Each have been responsible for working with the CAS Master Teachers in their area to promote and support relevant teacher engagement and CPD activities, with the aim of establishing effective and enduring local communities of practice involving Master Teachers and local CAS Hubs. In addition, many universities can provide support to schools and teachers throughout their region through their outreach programmes.

If there's something you would like to see provided in your area, do get in touch with them, their contact details are also available on the website.

Join in with #caschat every Tuesday

Every Tuesday night – from 8:00PM to be precise – the twittersphere positively lights up with CAS members sharing their expertise and practical advice under the guidance of the wonderful Simon Johnson (@clcsimon). Each week there is a different topic as listed in this #CASChat resource. You can also add your own topics there, and a series of questions about that topic are posed.

What follows, as the old guard will know well, is a hectic hour of terrific advice,shared from other CAS members, about how they approach that topic in their classrooms, plus some light-hearted chat, but all aimed at educating, engaging, and encouraging one another. It's a brilliant way of connecting with other teachers via Twitter and expand your own personal learning network.

> CAS IS NOT A SERVICE FROM WHICH YOU BUY PRODUCTS; IT IS A COMMUNITY, A NETWORK, OF FELLOW PROFESSIONALS LOOKING TO SUPPORT EACH OTHER

CAS is not a service from which you buy products; it is a community, a network of fellow professionals looking to support each other. It's a 'gift economy' and, in the spirit of Computing At School, I encourage all teachers to connect with their local CAS communities to find out what you can give to support your colleagues for the benefit of your pupils. "There is no them, only us!" (HW)

BCS CERTIFICATE IN COMPUTER SCIENCE TEACHING

The BCS Certificate in Computer Science Teaching provides certification which is tailored to the individual needs of computing teachers

STORY BY Nadine Vaillant Hill & Andrew Csizmadia

The Government's digital strategy (DfE March 2017) recognises the fact that the UK's future economic prosperity depends on us addressing the growing digital skills crisis. Schools are embracing the computing curriculum introduced in 2014; however, the national shortage of computing teachers is growing. This means that schools often need to turn to teachers of other subjects to help deliver the computer science elements of the curriculum including teaching the subject at GCSE. Schools and teachers can access a range of support, training and CPD from various organisation such as Computing At School and the Raspberry Pi Foundation, but how do they know their knowledge of the subject is at the standard to teach their GCSE students?

The BCS Certificate in Computer Science Teaching (the Certificate) was conceived by Sue Sentance, Senior Lecturer in Computer Science Education at King's College London, who recognised that teachers give up their time "developing their computing subject knowledge and a mechanism was needed to give recognition and "certify" teachers."

Sue and the team that developed the Certificate recognised that serving teachers are very busy, and they felt it was really important that the Certificate was "rigorous, but not onerous" and made a difference to teaching practice and student learning. The certificate was developed to have "a focus on pedagogy and formative assessment of drafts, and gave teachers an opportunity to work on useful projects as well as provide certification." Thus, the Certificate provides teachers with certification and recognition for their professional development.

Structure and Support

The two versions of the BCS Certificate in Computer Science Teaching (Primary and Secondary allow teachers to reflect the school settings they work in. Teachers have up to a year in which to complete the three parts that make up the Certificate. A bespoke, secure Certificate Learning Environment has been developed and continues to be updated to support teachers with the guidance and resources to enable them to complete the Certificate successfully.

As teachers work towards the Certificate they are supported by their individually assigned e-assessor, who will support them in developing their evidence for the Certificate. The Certificate has been designed and developed around a model of formative assessment. This means the e-assessor provides meaningful and constructive feedback on plans and drafts at each stage to help teachers to improve and refine their work instead of simply submitting it all at the end, when they believe they have achieved the standard required for the Certificate.

Components of the Certificate

As indicated the Certificate consists of three parts, as shown, which can be completed in any order. **(Fig.01)**

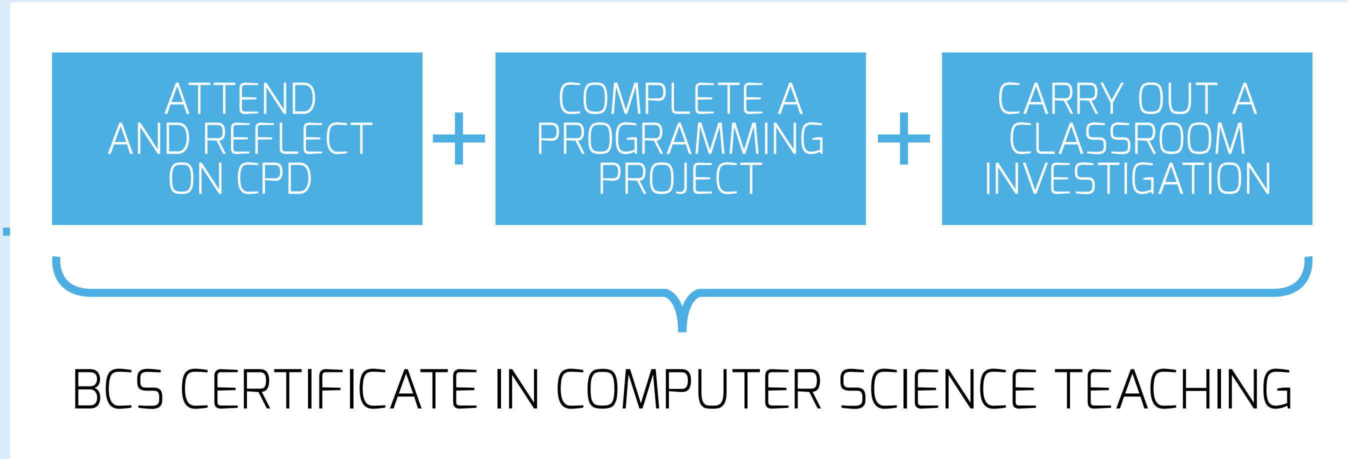
01 Reflection on professional development

To pass Part One of the Certificate, a teacher is required to show evidence of having undertaken at least 20 hours of eligible CPD within the previous two years. The evidence is recorded in a professional development log which includes their reflections on the impact of the CPD on their own learning, on their teaching of computer science and for their learners. As with all evidence and communications, it is submitted via the secure Certificate Learning Environment. Evidence can be drawn from a variety of sources, e.g. CAS Hub Meetings, Picademy training and online learning.

02 Programming project

The second component of the Certificate has been designed to allow a teacher to design, develop and submit a

Fig.01 Overview of the three components of the BCS Certificate in Computer Science Teaching



working program. Secondary teachers are required to produce a program using a text-based language of their choice. Whilst a primary teacher can produce a program using either a block-based language or a text-based language if they prefer. Instead of writing small snippets of code, teachers demonstrate their ability to produce a complete working project by using a range of programming techniques.

Projects that teachers have produced include:

- Text-based adventure game (Secondary)
- Demonstration of encryption techniques (Secondary)
- Numeracy quiz (Primary)

Teachers submit their proposal of the program they intend to develop and receive constructive feedback from their e-assessor regarding the suitability, feasibility and complexity of the program proposed. Teachers need to act upon the feedback and develop a program which meets their design brief. Teachers will receive feedback on their first draft suggesting any improvements before submitting the final revised program.

03 Classroom investigation

For Part 3, the teacher needs to do on a small-scale classroom investigation with reflections and a report. This will focus on some aspect of teaching the computer science elements of the computing curriculum. As with Part Two, teachers submit a proposal for their investigation, act upon the constructive feedback they receive from their e-assessor before conducting their investigation using the most appropriate research methods and tools. A short draft report is provided for feedback before they submit their final report.

Teachers have chosen investigations such as:

- How can students successfully progress from Scratch to Python?
- The importance of storytelling in teaching programming
- Does paired programming help students with debugging?

Exemplar materials produced by teachers who have completed the Certificate are available to help teachers at each stage of the Certificate.

Support for achieving the Certificate

Following the established principles of good CPD, the Certificate was designed to support teachers from their own starting points and to achieve the learning objectives that they set for themselves. Consequently, teachers can choose to achieve the Certificate independently, drawing on their existing subject knowledge and attendance at relevant courses.

Alternatively, the Certificate team offer online courses to guide and support teachers for both Part 2 and Part 3. Teachers can enrol on the courses at any stage of working towards the Certificate, even if they began with the plan to achieve the Certificate independently. This helps when busy teachers find they need the structure attending a course provides, or they need to supplement their subject knowledge These online courses are delivered by subject matter experts, such as Dave Ames, Duncan Maidens and Jane Waite.

The Certificate is based upon a model of experiential learning.This approach allows a teacher to realise the benefit of the CPD they have attended; their own personal study by putting what they have learnt into practice by creating a functional project, and investigating an aspect of teaching computer science in the classroom. (HW)

SUMMARY

- For further information: **www.bcs.org/teachingcertificate**
- An evaluation of the Certificate by Sue Sentence and Andrew Csizmadia is available at: **helloworld.cc/2CGU4r3**
- "My assessor was brilliant and always on hand to answer any of my questions. She was also very good at marking/responding within a few days of me submitting my assignment." **Nicola Hancock (Secondary Teacher)**

PICADEMY

GETTING HANDS ON WITH COMPUTING

Picademy is the Raspberry Pi Foundation's in-person training programme, reaching over 1500 educators worldwide. Here we explore the approach that has made Picademy successful

STORY BY James Robinson

At the Raspberry Pi Foundation we fundamentally believe that Computing and Digital Making is for everyone, regardless of age, gender or background. It's important that everyone has the opportunity to learn these empowering skills. However, like learning any new skills, people need time, confidence, and a context in which to learn them. This is precisely what we aim to deliver through Picademy, our free face-to-face training programme.

Picademy runs throughout the year in locations across the UK and North America and welcomes educators of all backgrounds to participate. Educators that join us for the two-day event graduate as 'Raspberry Pi Certified Educators' and with renewed confidence as makers and programmers.

Over the course of the training we explore computing through engaging contexts including music, electronics, robots, and sensors. These contexts provide concrete and challenging experiences through which to develop and explore the abstract concepts associated with computing. Whilst we use Raspberry Pi computers to deliver much of the training, it's not about the device but the approach and the importance of hands-on, learner-led experiences. We want learners everywhere to have exciting Computing and Digital Making experiences, the platform or technology used isn't important.

Arguably the most important part of the Picademy experience is the second day where our educators are challenged to apply learning from day one. We support participants in developing their own project to consolidate and extend their skills, decomposing their ideas, and iterating as they go. Through this project they collaborate, show resilience to the inevitable (and valuable) failures and experience computing from their learners' perspective.

Hearing educators share their ideas, failures, and reflections is always a highlight for me, as is hearing their aspirations for the next steps they will take beyond the event.

Educator community

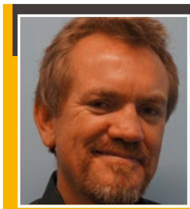
Being a Raspberry Pi Certified Educator (RCE) means more than just having participated in a Picademy event. Graduates become part of an active global community, impacting their learners, improving their own knowledge and sharing with other educators.

RCEs have gone on to start extracurricular activities, introduce new schemes of work, lead awe-inspiring projects, train their fellow educators and speak at local or national events and much more.

> WE'RE ALWAYS LOOKING FOR NEW WAYS TO SUPPORT OUR EDUCATOR COMMUNITY AND WELCOME NEW MEMBERS

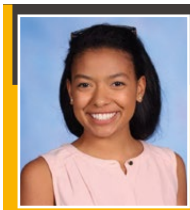
We're always looking for new ways to support our educator community and welcome new members. We learn a lot from the community through our annual survey (**rpf.io/survey**), which led to us offering online training in 2017. As we prepare for our 2018 programme of training, we're continuing to listen to educators and will provide more ways to support both educators and their learners.

For more information about Picademy and to apply for one of our 2018 events, please visit **rpf.io/train**. (HW)



MARK NORWOOD

Being an RCE means being part of a fantastic wider Pi community. This is a two-way thing; you have an amazing, friendly support system on tap which will very quickly answer even the silliest questions, while hopefully you contribute in turn by sharing what you're doing with the community, thereby inspiring and helping others.



VENUS MONTES DE OCA

After two phenomenal days of thinking, hacking, creating, planning, trying, failing, and making, I earned my title as a Raspberry Pi Certified Educator. Being an RCE, I find myself among individuals who are excited to revolutionise the way we use technology in the classroom and beyond.

CREATING MIDDLE AND HIGH SCHOOL COMPUTER SCIENCE TEACHERS IN JUST FIVE DAYS

Almost 800 teachers spent 5 days of their summer with Code.org to learn how to teach a subject many of them have never taught before.

STORY BY Dani McAvoy

Summer 2017 marked the launch of Code.org's newest curriculum: CS Discoveries. The inauguration of this introductory computer science course was special, not just because it was rolled out, but because it successfully scaled from 40 teachers piloting the curriculum in the spring, to 800 teachers prepared to bring the course to their students in the 2017-18 academic year. How does Code.org find 800 educators who can teach computer science in middle and high school classrooms across the United States? We make them!

What we offer teachers

The Code.org Professional Learning Program gives teachers the tools they need to succeed with minimal to no background in Computer Science (CS) through extensive curriculum resources and professional learning opportunities organized by course. For CS Discoveries, the professional learning starts with a five day in-person training called TeacherCon during the summer before teachers head back to school to teach.

During TeacherCon, teachers and facilitators spend five days immersed in conversations about the curriculum and teaching practice. Participants prepare to teach a lesson from the curriculum to their peers in a non-threatening environment where they can try out new teaching practices. Through the process of experiencing lessons taught by their peers, teachers build empathy with the learner experience, feeling what it will be like for their students to do these activities.

The professional learning acknowledges that one of the biggest assets teachers can leverage is each other. In a room of 30 teachers, participants are bound to learn new ideas and strategies from one another, rather than the person standing at the front of the room. Teachers discuss challenges they will face in their classrooms, and share teaching practices with each other. As one teacher put it, "I liked networking with teachers and learning from them. Everyone has great ideas and new strategies, etc. I wouldn't have got this breadth of information and ideas otherwise." Throughout the week, discussions and activities range from the curriculum to student recruitment, to engaging the broader community of stakeholders in their area. Throughout all of this, feedback, and adjusting to participant needs is an essential part of the TeacherCon experience.

Why five days?

In an ideal world, where time is unlimited, we might wish to work with teachers for months before sending them into class, but the realities of classroom teachers' commitments make that impossible. So why start the Professional Learning program with a five day TeacherCon rather than trying to prepare teachers in less time? Because change takes time.

We encourage participants to explore new teaching practices . CS Discoveries utilizes inquiry learning, and certain teaching philosophies which are new to teachers. To really change teachers' practice takes time and exposure to the benefits.

"Allowing students to learn through personal discovery had always been difficult for me to allow as a teacher. The [lessons] have put me in the student role and I now see how valuable student self-discovery with CS is to the process."

We aim to build confidence. Many teachers express feeling overwhelmed on Monday of TeacherCon, especially those who are new to CS. Through experiencing lessons as learners, their confidence grows.

"My favourite part of TeacherCon was gaining content knowledge and curriculum navigation, because I feel like I have enough information to advance on my own. Previously, I didn't even know where to start and felt like I didn't have enough content knowledge to even try."

Trust is essential for real conversations. The tough conversations, that really empower participants to grow as teachers, require trust and that takes time to build. This also develops a community that supports one another throughout the year.

"This truly has been one of the best professional development sessions that I have attended. Although I am mentally stretched, I was engaged, supported, and affirmed the entire week."

Beyond the week

The support teachers get during the five days of TeacherCon doesn't end when they leave at the end of the week. In addition to in-person local workshops throughout the year, Code.org treats its 'Curriculum as a Service,' which means we actively and constantly build new supports for teachers. (HW)

BECOME A PART OF IT!

Interested in becoming part of this national community expanding CS? Join the community at **forum.code.org!**

DONORS

All of the work we do at Code.org is made possible by our generous donors who believe in expanding access to CS education. We and the teachers really appreciate the generosity!

PROFESSIONAL **DEVELOPMENT** GETS PERSONAL

Gary Stager walks us through his own personal approach to professional development

STORY BY Gary Stager

Seymour Papert once said, "Schools are such bad places for children to learn because they are bad places for teachers to learn." Teacher professional development needs to be in a context, and pedagogical style, that models our highest ideals for educating children. Good teaching is good teaching.

My approach to professional development over the past thirty-five years is based on four perspectives.

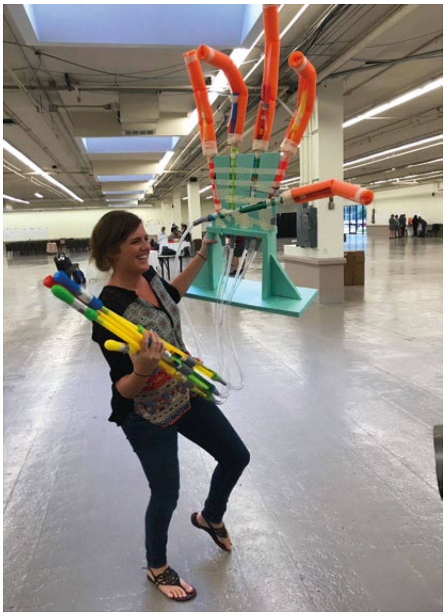
- Professional development should directly help teachers better serve students, not complete clerical chores. Learning to master a new technology is an often complex or frustrating process. That process is compounded when we associate learning to use technology with something you dislike or resent doing.

- We should treat teachers with respect and dignity. They are competent and therefore our PD efforts do not need to treat them like infants of felons. 'Training' is for circus animals, not professionals. Teachers should not require training in using Google or an iPad, something every toddler has mastered pretty well by now without government funding.

- Given the scarcity of time and resources found in schools, professional development has an obligation to blow an educator's mind and motivate them to continue learning long after the PD ends.

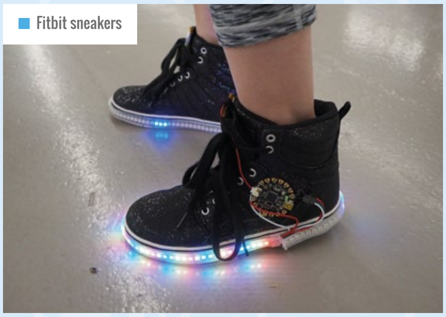
- The project should be a teacher's smallest unit of concern. The same goes for PD leaders.

We should treat teachers with respect and dignity



No training required

Constructing Modern Knowledge



Fitbit sneakers

Colour-changing glove

Popular children's author Peter Reynlds rides the cardboard tricycle

A decade ago, I created Constructing Modern Knowledge, a summer institute where educators from around the world gather to learn about learning by learning with cutting-edge technology and a mountain of other materials, while working on personally meaningful project development, and without either coercion or formal instruction.

All of my work is based on the Piagetian principle that knowledge is a consequence of experience. Each CMK begins with participants sharing ideas for what they wish to make, quickly followed by four days to work on such projects. Each year, CMK participants confirm my hypothesis, "A Good Prompt is Worth 1,000 Words." With a good prompt or interesting challenge, appropriate materials, sufficient time, and a supportive culture, including a range of expertise, people are able to exceed their wildest expectations.

> ALL OF MY WORK IS BASED ON THE PIAGETIAN PRINCIPLE THAT KNOWLEDGE IS A CONSEQUENCE OF EXPERIENCE

Over the past few institutes, dozens of remarkable projects have emerged. These include: Fitbit sneakers that light with each step and perform a dance show when your step goal is reached; a mechanical and digital sculpture capturing the poetry of wind; automated greenhouses and systems to water thirsty plants; an adult-size cardboard tricycle; a four-person Chinese dragon, complete with eTextiles and microcontroller-based eyes; a helium balloon-powered drone; and working versions of Pokémon Go, including information about the local community and even our institute space. (HW)

TIPS FOR PD SUCCESS

- Ask participants to take off their teacher hats and put on their learner hats!
- Expect the impossible, and your students will surprise you.
- Whimsy, beauty, playfulness, and mystery are powerful contexts for learning.
- Focus on powerful ideas, not step-by-step mechanics.
- Offer maximum choice in projects and processes.
- Establish an absence of coercion. Operate under the assumption that your students want to be there. "Nothing beautiful can ever be forced." - Xenophon
- Supply sufficient materials and time. Quality work takes time and you don't want people waiting around for materials.
- Papert teaches us that the best learning results from hard fun.
- Less us, more them. Provide a minute or two of instruction, suggest a prompt or challenge, and then shut up. The more agency one can bestow upon learners, the more they will accomplish.

(Hello World)

HOW TO SUBSCRIBE

- **Visit us online:**
  helloworld.cc/sub1year
- **Call our subscriptions hotline:**
  +44(0)1202 586848

Not a UK-based educator?

- Buy any issue for £6

Visit: helloworld.cc/buyissue

- Subscribe from £15 for 3 issues

Visit: helloworld.cc/sub1year

SUBSCRIPTION FORM

YES, I'd like to subscribe to **Hello World magazine**!

This subscription is: ☐ For me ☐ A gift for someone

HW#3

YOUR DETAILS ☐ Mr ☐ Mrs ☐ Miss ☐ Ms

First name .... Surname ....

Address ....

....

Postcode .... Email ....

Daytime phone .... Mobile ....

Are you a UK-based educator? ☐ Yes ☐ No

GIFT RECIPIENT'S DETAILS ONLY ☐ Mr ☐ Mrs ☐ Miss ☐ Ms

First name .... Surname ....

Address ....

....

....

Postcode .... Email ....

Is the recipient a UK-based educator? ☐ Yes ☐ No

If you're giving Hello World as a gift, please complete both your own details (left) and the recipient's (right).

Your Hello World subscription will start from issue #3, unless you specify the issue number here: ....

RETURN THIS FORM TO: Hello World Subscriptions, Select Publisher Services Ltd, PO Box 6337, Bournemouth BH1 9EH. *We'll be in touch!*

☐ Please tick this box if you DO NOT want to subscribe to the Raspberry Pi Foundation Education newsletter (no spam).

CARRIE ANNE PHILBIN DIRECTOR OF EDUCATION AT THE RASPBERRY PI FOUNDATION

THERE IS NO THEM, ONLY US

How communities of computer science education practice are the future of CPD

Teaching can be a very isolating experience. I remember having days at school teaching ICT and Computing where a jam-packed timetable and some speedy site changes meant that I did not speak to another adult all day! Even when time was provided for working with department colleagues, we would end up concentrating on grading or examinations and not really communicating our thoughts, feelings, and frustrations about our subject. Can you imagine anything more isolating in a profession that encourages observation, self reflection and collaboration for improvement? Continuing Professional Development (CPD) isn't just about what you know, or even what you know about teaching it. It should be an investment in the personal well-being of the educator which means time for that individual to learn and apply new ideas, but also the opportunity to build a collaborative and supportive personal learning network. Luckily, there are lots of opportunities for meeting other educators like you, thanks to communities of practice in both the formal and informal CS education sphere.

Getting out and about

When the ICT curriculum was disapplied in England in 2011, I wanted to learn more about incorporating computer science into my lessons, but I quickly realised I did not know where to start or who to speak to. It became clear to me that, to get the support I needed, I would have to leave the safety of my classroom where I was pretty confident, and get out and about and learn from others.

A casual Google search back then provided me with three possible groups where I might find what I was looking for: a Raspberry Jam, the UK Python developers conference (PyCon UK) and a Computing At School meetup, known as a Hub. One regional event, one national, and one local seemed like a good way to spread my bets. The thought of attending these three events filled me with fear. What if they discovered my secret, that I'm the classroom teacher that is getting all the bad press right now for being 'irrelevant'? What I found couldn't be further from my anxious imaginings: three different, but sometimes overlapping, communities of friendly individuals, who couldn't do enough to provide opportunities to connect and support.

The Python software community surprised me the most. I was one of a handful of teachers in an over-crowded room of professional software developers who attended PyCon UK that year. We were called out of the audience for all to see, and I died a little inside. But then, the keynote speaker started to talk about a Raspberry Pi computer and how this low-cost device had the potential

First PyCon UK educators event September 2012. Image courtesy of Nicholas Tollervey to improve the learning of my students, and things started to look up. After that, we were invited to a smaller break-out room where we were paired with developers, and I wrote my first lines of Python. The next day I found myself on a panel discussing formal education and, by the end of the day, I was presenting a five-minute lightning talk on the life of a computing teacher to the entire conference! The miraculous improvement in confidence over two days was solely the result of how I was embraced and supported by the Python community. I was even able to 'adopt a developer', who supported me through my personal practice outside of the conference, and all because I had the courage to look outwards rather than inwards for my CPD opportunities.

Sharing information and working together

Attending a national conference lit a fire within me, and provided the inspiration and confidence I needed to make change. It was like being able to see clearly for the first time in a long time. Attending a local meetup, on the other hand, was more akin to being hugged and given vast mugs of hot chocolate. Catching up with familiar faces working in institutions in your area and learning that they are struggling with the same problems as you is comforting. Working together, devising methods and tactics for resolving those problems before coming back together to share the successes and failures of those approaches is cathartic, useful, and above all else, fun. I've experienced these meetups through the Computing at School Network of Hubs and Regional Centres, as well as Coding Evenings, Code Club Meetups and Teach meets. You can find something local and useful to you through an online search, but remember that the key to success is what you put in, not just what you get out of it!

> I was one of a handful of teachers in an over-crowded room of professional software developers who attended PyCon

Get your hands dirty

There is a third category of event, which brings together educators, hobbyists, parents, and young people in one space to meet and learn from like-minded people in a friendly environment. Maker Faires and Raspberry Jams are two examples of this mixture of people. They can come in all shapes and sizes and include: workshops for beginners, drop-in sessions to work on your own projects, a show-and-tell, and often a marketplace to buy tools and kits. A Jam is a meeting of all kinds of people, and a great way to see families learning in informal contexts where there isn't a curriculum or set of standards that need to be followed. You can pick up ideas, speak to parents about their motivations for bringing their children, find out what is resonating, and make connections with knowledgeable hobbyists about the projects that they've created. I've found lots of practical applications of computer science at these events, sparking creative and engaging lesson ideas

Education Panel at PyCon UK 2017. Photo taken by Mark Hawkins at PyCon UK 2017 and published under CC-BY 2.0 license

for back in the classroom, and the best part is knowing that I can call on a community of individuals to help implement it, rather than having to do all the work alone.

Conclusion

Continuing professional development is defined as a record of what you experience, learn, and then apply. It should not be an annual appraisal tick box exercise, nor should attending a one or two-day course a year be sufficient. I'd encourage every teacher of computing and digital making, from rookie to expert, to seek out and join different educational communities and share best practice with each other. Working with others to solve problems in a new and rapidly improving subject area is much easier, and a lot more fun that trying to tackle them on your own. As we say in the Computing At School community: There is no them, only us!

- National Python conferences take place in countries all over the world so, wherever you are, you should also be able to access this amazing community. www.python.org/community
- Find a Raspberry Jam near you - www.raspberrypi.org/jam
- Join the Computing At School community wherever you are in the world - community.computingatschool.org.uk
- Attend a Maker Faire - makerfaire.com (HW)

Carrie Anne Philbin is Director of Education at The Raspberry Pi Foundation, a Computing At School board member, author, and YouTuber.

ARE THERE PROGRAMMING PARADIGMS?

It seems that there are lots of approaches to programming, but are they really so different?

STORY BY Greg Michaelson

In our young discipline, there have been continuous changes in programming practices, driven by rapid growth in the scale of software systems, and increasing sophistication in programming languages. Hence, it seems worthwhile to seek some organising principle to codify these practices, to enable us to assess their applicability, and to teach others how to do so.

A paradigm is a self-consistent, and often dominant, body of knowledge and practice. The idea was popularised by the American philosopher Thomas Kuhn, who studied how prevailing scientific paradigms come to be superseded by new ones, in the face of mismatches between what the orthodoxy predicts and observable reality. For example, in physics, the late 17th century Newtonian paradigm is a great model of how things behave at low speeds, but breaks down as speeds approach that of light. Thus, it was displaced by the early 20th century relativistic paradigm. As Kuhn observed, paradigm shifts involve fundamental, and typically strife-filled, changes in how the world is characterised.

By analogy with scientific paradigms, it's now common to refer to programming paradigms as if they were well-defined, alternative ways of doing programming, facilitated by corresponding programming languages. However, there seems to be a wee bit of confusion about what these paradigms are, and how they relate to each other and to programming languages.

Procedural-oriented, object-oriented and structured programming

For example, a widely adopted Computing syllabus distinguishes procedural-oriented (PO) and object-oriented (OO) programming, as well as functional programming, to which we'll return to below. I think that PO programming involves analysis, typically top down, to identify modular program units that can realised as subroutines, often parameterised for wider applicability. In contrast, OO programming requires the identification of classes of structured data manipulated by bespoke methods.

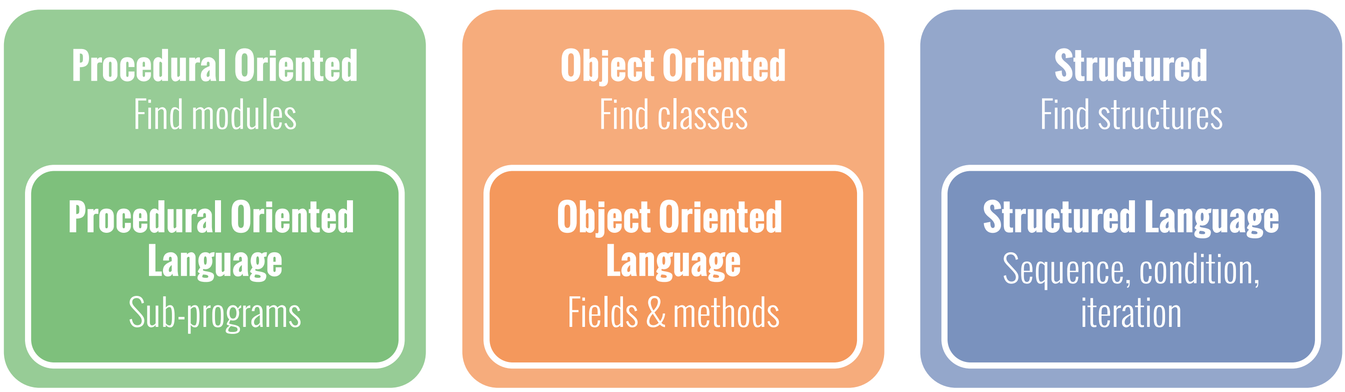
What about structured programming, which this syllabus doesn't distinguish? Maybe it's a separate paradigm, based on using sequencing, conditions, and iterations to compose assignments and I/O into programs. For sure, it's about the only approach that can be used by beginners who are learning to code in jigsaw environments, like Scratch or Alice, that offer pieces for the structured constructs. So maybe we have at least three distinct paradigms: see **Figure 1**.

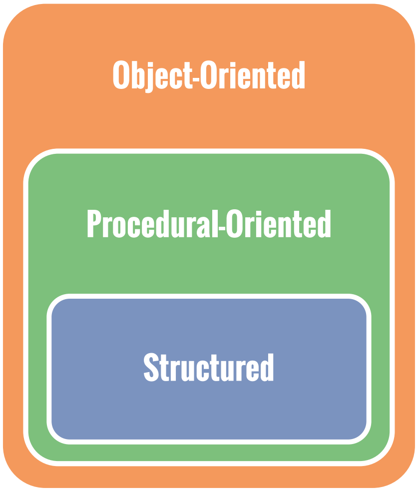
Then again, maybe structured programming is a technique rather than a paradigm, given that it's applicable to both PO and OO programming, to compose assignments and I/O into subprograms, and subprograms into larger subprograms. So perhaps structured is a subset of PO and OO.

Furthermore, once classes have been identified in OO, maybe it isn't so different to PO. And the insides of classes looks suspiciously like a top level procedural programs, where fields are treated as if they were global variables. So maybe PO is a subset of OO.

So, what about the idea that programming paradigms have corresponding programming languages, with paradigm appropriate

Figure 1: Different paradigms have different properties





■ **Figure 2:** Imperative paradigms are nested

constructs? Thus, we might expect that structured languages would offer sequences, conditions, and iteration, procedural languages would have subroutines, and object-oriented languages would provide classes. But, once again, given that PO and OO languages both offer subprograms, as well as all the structured constructs, it looks like structured is a subset of PO, which is a subset of OO: see **Figure 2**.

Functional programming

Still, functional programming (FP) seems markedly different. Pure FP languages are certainly well characterised. They lack modifiable variables, so there's no assignment, only function calls to bind formal parameters to actual parameter values. And it's common to distinguish functional, or the wider declarative languages, from imperative languages based on modifiable variables.

In FP, it's not possible to sequentially or iteratively update variables. Instead, recursive function calls are used to create new instances of formal parameters, bound to actual parameters values, formed from old instances. Additionally, pure FP languages support higher order functions, where functions may be passed to functions as actual parameters and returned as final values.

However, it's not so clear what the FP paradigm looks like, as there's no distinctive discipline of doing functional programming, other than programming in a functional language. And, while it's much easier to do FP in a functional language, it's perfectly feasible to program in a functional style in languages that aren't functional. Thus, pretty well all PO and OO languages support recursion, and cunning use of jump tables or objects can give the effect of both passing and returning functions.

Contrariwise, FP is inherently procedural, in terms of the key roles of identifying and composing modules realised as functions. Similarly, FP is inherently structured: function composition is equivalent to sequencing, and recursion to iteration. And all FP languages have conditional constructs. So, maybe FP and OO share PO and are structured: see **Figure 3**.

Paradigm and language

Now, it may well be easier to build software in one language rather than another, depending on the constructs that the languages offer. Nonetheless, it would seem unremarkable to realise a PO design in Java. And while it might appear forced to realise an OO design in Pascal, it's certainly doable. So, maybe the connection between paradigm and language isn't as tight as at first sight.

Furthermore, even if these different languages really do support alternative approaches to programming, there's something badly wrong if the resulting programs don't all compute the same outputs from the same inputs.

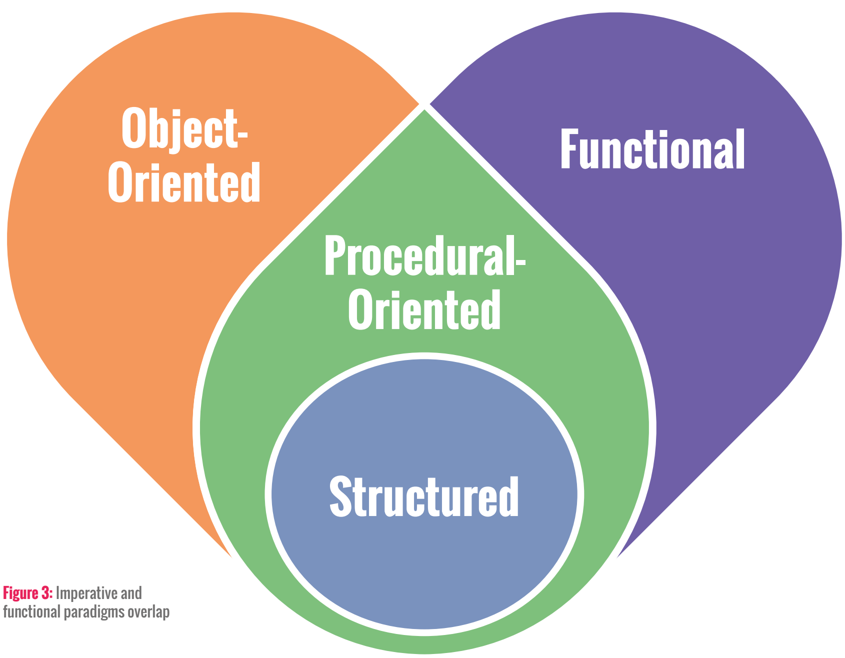
Indeed, as Church and Turing hypothesised 80 years ago, all models of computing turn out to be equivalent. A fundamental implication is that a program written in any one programming language can, in principle, be written in any other. Pragmatically, everything boils down into machine code.

Computational Thinking and programming paradigms

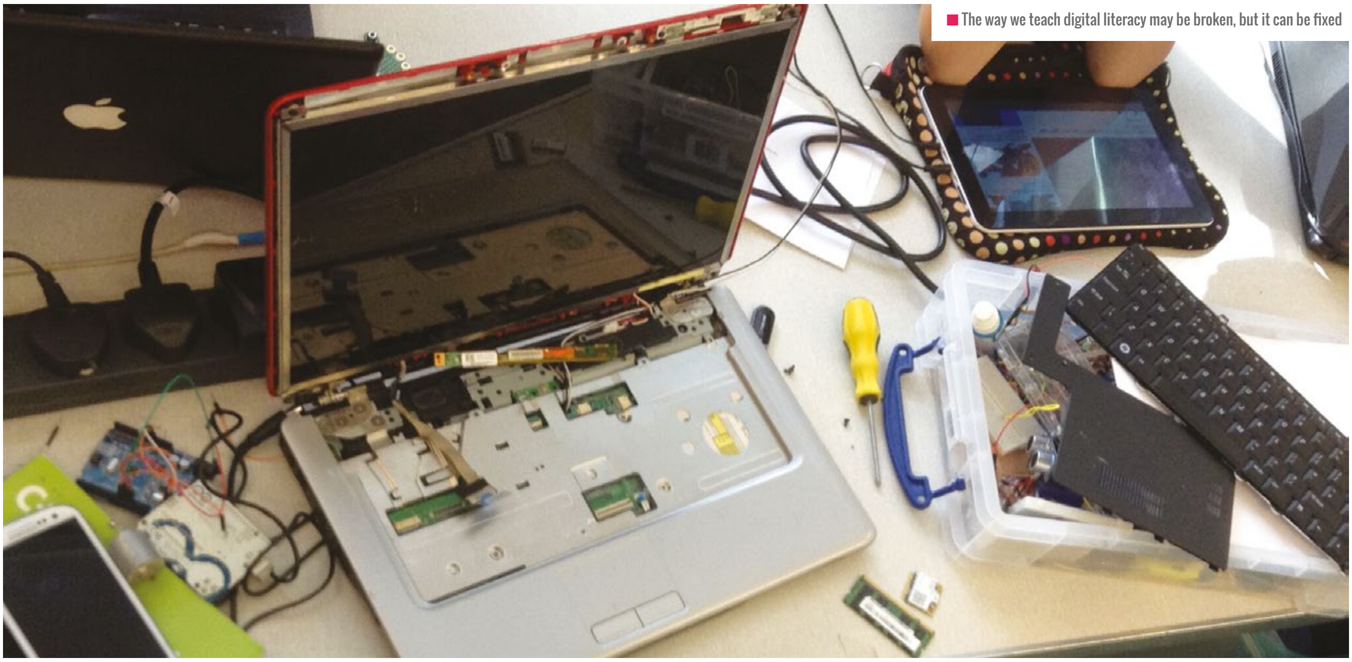
Thus, I'm no longer sure how to characterise a programming paradigm, or, assuming that there are paradigms, where one stops and another starts. Furthermore, the paradigms we've explored above aren't competing; rather, they're strongly connected. Unlike scientific paradigms, programming paradigms don't seem to be clearly disjoint.

Instead, suppose we return to Computational Thinking (CT) as a discipline of problem solving, and recall the stages of decomposition, pattern identification, generalisation, and algorithm construction. Then, I think that identifying classes or modules or functions are primarily to do with decomposition, and structured programming is about algorithm construction, from subprograms to entire programs. I also think that programming, in the sense of coding in a specific language, comes at the very end.

Now, if we start off with the intention of building programs in a particular language, that will certainly affect how we formulate our solutions from an early stage. Nonetheless, maybe CT itself is the paradigm for programming? (HW)



■ **Figure 3:** Imperative and functional paradigms overlap

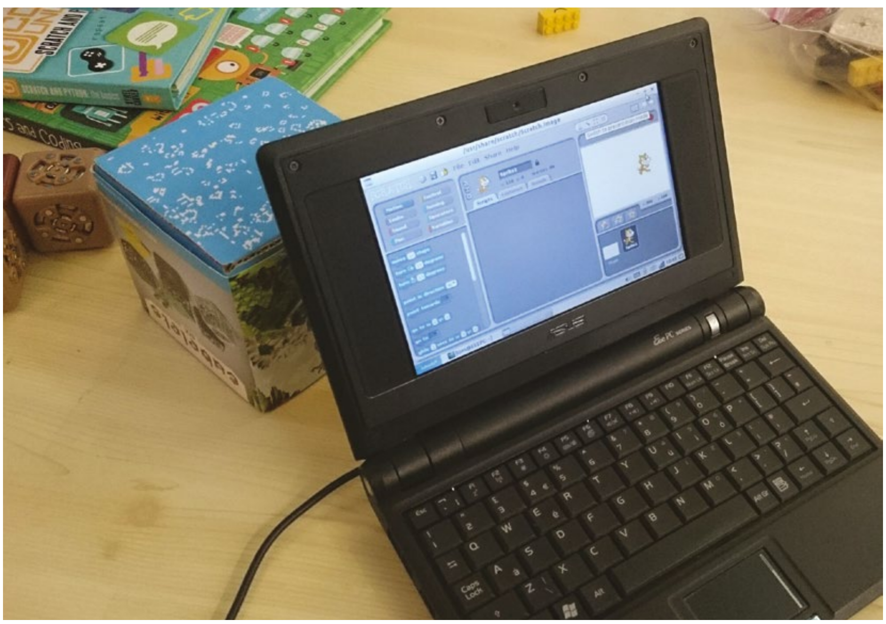


The way we teach digital literacy may be broken, but it can be fixed

DIGITAL LITERACY: LOST IN TRANSLATION?

Programming teaches valuable lessons, but it isn't the whole story. Does an exclusive focus on programming in schools fail to give students the knowledge and skills they need to become digital citizens?

STORY BY Benjamin Wohl



Instead of just teaching kids how to code, digital literacy needs to give them the tools to become digital citizens

Many of the early proponents of bringing more computer science to schools were inspired by the idea that, first and foremost, young people needed to be digitally literate. The widely cited Royal Society report *Shut Down or Restart?* made it clear that teaching digital literacy is essential before pupils move onto more complex concepts of computer science. But what does it mean to be digitally literate?

Although the term 'literacy' is strongly related to the specific skills of reading and writing, it also carries with it a connotation of being well-educated in the core texts of a culture. Being literate is not just about having competency, but is also about having a breadth of core knowledge and having critical reasoning skills. Looking at the English Computing Curriculum, it seems that

Teaching digital literacy is about more than just e-safety - it is about giving kids access to the right tools to become confident digital citizens

digital literacy is simply being boiled down to the ability to use computer technology competently and safely.

From fake news to adblockers

Digital literacy should not be reduced to concerns about e-safety, but rather needs to involve teaching young people to become capable digital citizens. Rather than being able to write rudimentary code in Python or C++, a digitally literate individual should be able to distinguish between a personal blog, a BuzzFeed article and a fan wiki, and know when to use each as an information source. They should be able to navigate and engage confidently with a range of online communities and platforms, negotiating fake news and ad-blockers. They should have an awareness of community-generated software, including common open-source platforms like LibreOffice or Minecraft mods, and know how to use these safely. These are also the skills young people want to learn, and will be able to apply in their lives. Many young people are already actively engaged in these worlds, and this makes the need for teaching digital literacy even more urgent.

> THEY SHOULD BE ABLE TO NAVIGATE AND ENGAGE CONFIDENTLY WITH A RANGE OF ONLINE COMMUNITIES

The keys to participation

In the past, being literate has been as much about understanding information as accessing it. Digital literacy continues to be about having the technological know-how, as well as the critical thinking skills to participate in online conversation and online communities. Young people should not be able to leave education without the thinking skills they need to be digital citizens, or the language needed to discuss the ethical debates surrounding digital culture.

Fundamentally, literacy is about being part of a community. It is about having access to the technologies that contain the collective memory of that community. Teaching digital literacy is about giving young people the keys to being part of a digital world, knowing how to navigate and use the tools of the internet. The biggest difficulty we face as educators is that these rules and norms are still being written, and sometimes it can seem easier just to teach kids to code. (HW)

DIGITAL LITERACY IN THE ENGLISH COMPUTING CURRICULUM

- Understand a range of ways to use technology safely, respectfully, responsibly, and securely; including protecting online identity and privacy; recognising inappropriate content, contact and conduct; and knowing how to report concerns
- Use search technologies effectively
- Understand the opportunities [networks] offer for communication and collaboration
- Be discerning in evaluating digital content
- Use technology safely, respectfully and responsibly; recognise acceptable/unacceptable behaviour; identify a range of ways to report concerns about content and contact
- Recognise common uses of information technology beyond school
- Use technology safely and respectfully, keeping personal information private; identify where to go for help and support with concerns about content or contact on the internet or other online technologies

Ben is a PhD candidate at Lancaster University's HighWire Doctoral Training Centre. He is researching the influence of digital economy policy on the English Computing Curriculum, from looking at hands-on approaches to teaching computing at KS1, to conducting qualitative studies in KS3 classrooms.

Sevenoaks Preparatory School students using Code.org

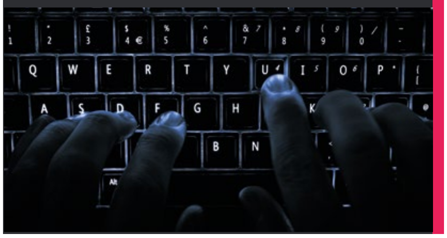
COMPUTATIONAL THINKING USING CODE.ORG

Code.org offer a superb set of free resources suitable for a wide age range. This article will explore the variety of courses on offer to help in the teaching of computational thinking

STORY BY Ronan McNicholl

GET STARTED IN 4 EASY STEPS

- Visit **Code.org** and choose 'sign in' on the top right-hand corner
- Create an account and sign up using an email address
- Add your class by clicking 'create a new section' and a sign in method
- Share the course link with your students via downloadable login cards

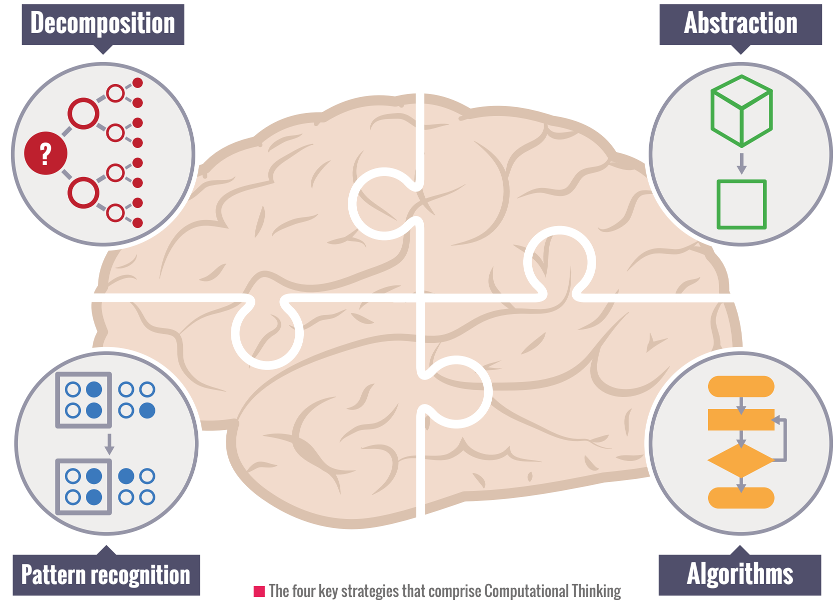


Code.org is a non-profit organisation launched by Hadi and Ali Partovi in 2013, to help promote the development of computer science in education. It receives donations from various sources, including large technology companies, and is now a thriving educational community.

Teachers can easily set up free accounts and assign courses for students to complete. This can help teachers to successfully deliver key objectives from the computing programmes of study across all key stages. A range of computational thinking concepts are covered in these courses including decomposition, pattern recognition, abstraction, and algorithms, as well as broader computer science objectives such as sequence, selection, repetition, and debugging. A great feature of the platform is the ability for both students and teachers to view and track progress. This is possible as the solution to each puzzle is stored in terms of accuracy and code quantity. This offers a great opportunity to promote assessment for learning and obtain summative assessment data. Another advantage is the built-in gamification. Each puzzle is comparable to a game, and students are therefore self-motivated to progress to the next stage or level. To further encourage students to use the platform, Code.org have partnered with popular brands such as Angry Birds and Minecraft. The courses incorporate a variety of such themes. Periodic videos, featuring computer scientists and engineers, help to add another level of engagement with the resource.

What is computational thinking?

Computational thinking is the process of applying mental strategies to solve problems. Fundamental to computer science, computational thinking helps create new algorithms which are then



■ The four key strategies that comprise Computational Thinking

HOUR OF CODE

Code.org run an annual worldwide Hour of Code event. This year it will take place from 4-10 December and is open to all schools, with new themes unveiled each year. Previous examples have included one-hour courses based on Star Wars, Frozen, Moana, and Minecraft. The aim is to promote computer science for all. Courses are typically available in over 30 languages. Promotional materials including sample letters, stickers, and posters are all available on the **Code.org** website. A database of industry professionals is also stored, and these individuals can be approached to appear at schools as motivational guest speakers.

implemented within mobile apps, websites, games, and software. However, the benefits of computational thinking are not confined to computer science. Learning to apply computational thinking can benefit everyone in society, and it is important that all students have the chance to learn these strategies at school. Four key strategies comprise computational thinking: decomposition, pattern recognition, abstraction, and algorithms. Let's now look at how Code.org helps in the teaching of computational thinking.

> IT IS IMPORTANT THAT ALL STUDENTS HAVE THE CHANCE TO LEARN THESE STRATEGIES AT SCHOOL

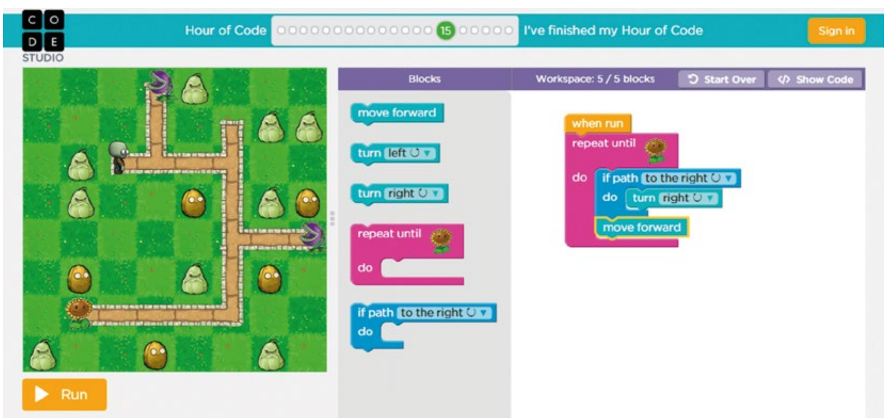
Computational thinking using Code.org

Pattern recognition involves looking for patterns or trends within a problem. A lot of the Code.org puzzles are pattern-based and teachers can easily model the process of identifying these patterns and then progress to demonstrating how to solve them more efficiently by using loops.

Algorithmic thinking is fundamental to solving puzzles in Code.org as each solution is derived by sequencing, selecting, or repeating blocks of code. Modelling the process of adding these blocks, one step at a time, provides an excellent opportunity to demonstrate algorithmic thinking.

Decomposition is the process of breaking a complex problem into a smaller subset of problems which can then be solved individually, and eventually lead to an overall solution. When using Code.org, a teacher can demonstrate this strategy by building a solution one block at a time and explaining how each step helps solve the overall puzzle.

Abstraction can be taught explicitly in more complex puzzles by showing how functions can be reused in more complex programs, thus promoting code reuse.



■ Pattern Recognition using Code.org

Which course?

Which course you assign to your students really depends on their prior level of experience with the platform. As a rough guide, course A from the CS Fundamental track is pitched towards Year 1 students, course B is aimed at Year 2 students, and this repeats all the way up to course F for Year 6 pupils. Secondary students with no experience of Code.org could take an accelerated CS Fundamental course, and then progress to the more advanced CS Discoveries course and eventually the CS Principles course. A lot of the activities in the more advanced courses cover the same computational thinking strategies, though at a deeper and more challenging level. (HW)

ENGAGING PUPILS IN STEM USING DRONES

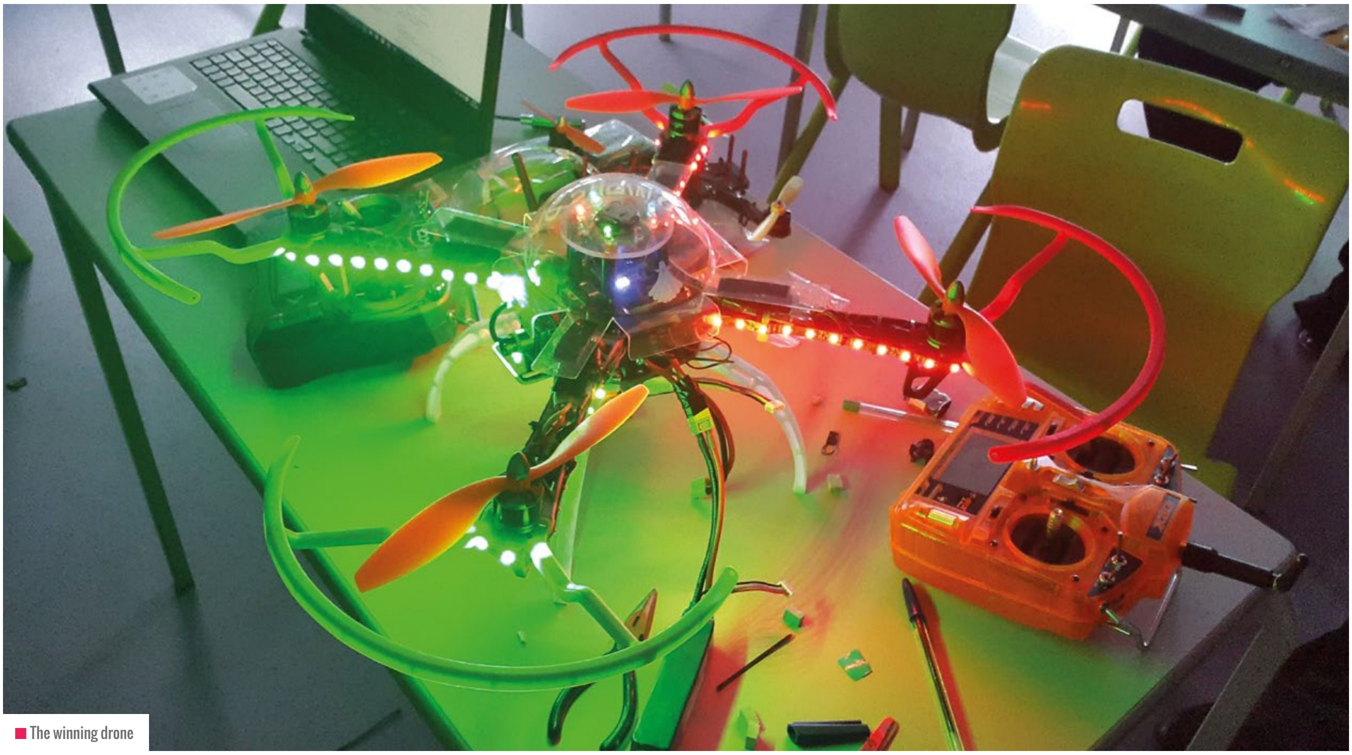
As the technology becomes more accessible and affordable, schools are using drones for a range of tasks...

STORY BY Neil Rickus & Penny Cater

The sale and use of drones has grown significantly in recent years, the devices now being used extensively as both a recreational hobby and a business tool. The industry is estimated to be worth £102bn by 2025 (**helloworld.cc/2FbDG0h**), and drone racing is even being considered as an Olympic sport (**helloworld.cc/2FdJaY9**). Within industry, Amazon is currently trialling their use for delivering goods (**helloworld.cc/2qFlAAw**), whilst the devices have proved vital for the emergency services following recent disasters (**helloworld.cc/2qDhMQn**), such as investigating the remains of the Grenfell Tower fire in London, or examining the aftermath of earthquakes.

As the technology becomes more accessible and affordable, schools are using drones for a range of tasks, from recording videos to capturing photos from trips (**helloworld.cc/2m6M5t3**). Many schools are taking this a step further and using programming environments, such as Tynker (**helloworld.cc/2CQBLAw**) or Apple's Swift Playgrounds (**helloworld.cc/2m6xkq1**), to give the devices instructions and develop pupils' understanding of computational thinking. Some devices, including those from Parrot (**helloworld.cc/2m8RC2m**), even have a 'Classroom mode' to increase safety when a number of drones are being programmed in close proximity (**helloworld.cc/2qFJfRi**). Some schools are also developing their use of the devices with dedicated clubs and/or as part of a 'maker curriculum'. Therefore, rather than just learning to fly and program the drones, they are designing, making, customising, and racing them. So, when CAS Master Teacher, Penny Cater, sent an email asking, "Would you like to come and visit one of our after-school drone sessions?" it was time to go and see her school's award winning work in action...

Penny says, "Drone club started at the Boswells School in Chelmsford (**helloworld.cc/2D9Tsbq**) two years ago, when we first entered the Raytheon Quadcopter Challenge (**helloworld.cc/2EmRbch**) and, during our initial competition which was held at Stow Maries Aerodrome, we came a close second. Our success in the competition led to us being invited by the Essex County Council STEM Skills Board to take part in 'build and fly' days at the Aerodrome. During



The winning drone

So much to learn! No need to drone on

the events, we were assisted by professional drone pilot Denis Stretton, who went on to become our STEM Ambassador, and the Boswells students learnt how to build 250 drones (250 is the size of the quadcopter from motor centre to motor centre, in millimetres). On returning to school, the students were so engaged I set up a drone club, and it was great to see the range of skills developed during our sessions – some students loved building and soldering, some simply wanted to hone their flying skills, whilst others loved programming/calibrating the software and working with the transmitters.

"In 2016, we reached the regional finals of the Raytheon Quadcopter Challenge and, with only two weeks between the initial competition and the final, the students spent every lunchtime and after school ensuring each team member could fly successfully. In order to work to each student's strengths, we structured the group like a Formula 1 team, with each person assigned a role they had a natural ability for. Using a range of drone equipment, the team's flying skills developed significantly, and it was pleasing to see how the motor learning was transferrable between different devices.

> IN ORDER TO WORK TO EACH STUDENT'S STRENGTHS, WE STRUCTURED THE GROUP LIKE A FORMULA 1 TEAM

There were five teams at the final and prizes were available for the overall winners, and for creativity and engineering. The team's theme was 'rainbow', in support of the Orlando nightclub incident in June 2016, and Claire Kuczma, computing teacher, also spent a lot of time supporting the students. Boswells' team was the youngest, with some pupils only in Y7 and, as part of their entry, the students had to prepare a presentation for the judges. Despite being asked a number of challenging questions, through using their knowledge and love for the drones, the Boswells team won and became National UK Quadcopter Champions 2016 for Raytheon! They were commended on their teamwork and knowledge of all aspects of drone engineering, building, and flying. Based on the learning from this competition, we purchased six '250' drone kits and the subsequent year's club attracted over 40 eager students.

"The drone club has been an incredible learning experience – the skills displayed and learnt include teamwork, patience, resilience, and perseverance, along with the practical skills of electronics, soldering, software calibration, and physical computing. I have thoroughly enjoyed running this club and could not have done it without the support of my team and the STEM Ambassadors. It has been great fun seeing the students learn and develop their skills, and given them opportunities that they wouldn't necessarily have got in KS3.

"Thanks to Ian Lewis, another STEM Ambassador linked with the club, we have now started to also examine 3D printing. Ian has kindly 3D printed micro drone frames for us, and students are learning how to build these devices from scratch to develop their own components.

"So, how can you start your own drone club? First and foremost, you need lots of passion and determination! During the club, I have three independent activities running, which helps promote inclusivity and ensures all students can benefit from the sessions. I run a drone simulator section, where students can practice flying with transmitters and get used to the controls and develop their motor learning, then I have an area for micro drone flying, where students set up a simple course and learn to control the drone's basic modes. Finally, there is a building section, where students have a kit and go through the build process.

"The drone simulator is being developed so that schools can link together and run inter-school regional competitions online. We have also connected with the Sydney Drone Racing School and hope to take part in international competitions next year.

"In addition to our STEM Ambassadors, our relationship with industry has been vital and we are now very lucky to have worked with Nigel Tomlinson, President of the Rotor Rush Academy and Ray Smith, from Drone Simulation, who have been incredibly supportive.

"If you want to buy your own kit, the best one I can recommend has been designed and developed by Chris Shaw at UKDS (**helloworld.cc/2qEgd4G**) and we were fortunate to have been consulted in its development. It comes with a comprehensive guide, which includes links to the curriculum and builds on Bloom's taxonomy. The kit also contains crucial guidance for non-specialists on safety, use of batteries, and flying restrictions.

"Want to know more? Contact @PennyCater on Twitter." (HW)

THE CODE'S NOT ALRIGHT

How do you decide when to teach a concept the 'right way' and when to simplify it?

STORY BY Laura Sach

It's a pretty common classroom task – you have some data of various types, and you want to teach students how to print the data out as part of a sentence, using Python. Here's my example data:

```
number = 1
person = "Scott"
```

The exam board wants students to know how to concatenate, so you might choose to teach the + operator, although it's a bit of a pain that the students will also have to put the spaces in the right place and remember to cast the integer to a string:

```
print("Thunderbird " +
str(number) + " is piloted by
" + person)
```

Alternatively, you might teach your students to separate the different parts with commas to avoid the casting issue. You either don't realise, or choose to disregard the fact that this isn't actually concatenation, it's taking advantage of the way Python's print() function works:

```
print("Thunderbird", number,
"is piloted by", person)
```

You could also choose to avoid the casting issue in a different way by teaching format(), but will this confuse the students?

```
print("Thunderbird {} is
piloted by {}".format(number,
person))
```

There are probably multiple other ways to achieve the same goal, each with their own benefits and drawbacks. Some people will argue vehemently for one or another (there's a long running debate about format() within the Raspberry Pi Education Team!), but which one is right?

Building a mental model

When teaching beginners, there is a conflict to resolve between teaching the way a more experienced programmer might approach a task, and teaching the learner to use a sub-optimal method, which either produces quick results or allows them to better access and build a mental model of what the program is doing.

As a teacher, I frequently shared resources online with the aim of helping others, and saving fellow teachers valuable time. Occasionally I would receive feedback on my resources from IT professionals – some was extremely helpful, and some was not. The helpful feedback had a collaborative feel, as if we both wanted to improve the outcome for children. The professionals helped me to improve my subject knowledge by providing a friendly suggestion for an alternative way to approach a problem where I hadn't realised that a different approach existed. This was extremely welcome, and I am indebted to people like David Whale, Martin O'Hanlon, and Andy Stanford-Clark (to name a few) for their generous mentoring.

However, I would occasionally receive feedback on my resources from IT professionals which was patronising and insulting. This type of feedback usually highlighted occasions when I had deliberately made a decision to teach a concept in a sub-optimal way, for a pedagogical reason. An experienced

What thought processes do learners go through to create a simple game?

developer can easily point out 'flaws' and suggest 'better' ways of doing things, but they often forget that, just because they are good at programming, that doesn't make them good at teaching programming. I've encountered developers who expect eight-year-olds to have mastery of concepts not introduced until A Level, and who forget that much of what they know is not basic, obvious, or picked up by telling a child once. This kind of feedback can really shatter a teacher's confidence, and I'd encourage developers to think about a teacher's reasons before commenting on 'bad practice'.

No short cuts to failure

It is sometimes even difficult for the teacher to take off their experience goggles and notice when they are short-cutting a learner's thinking process. Here's a recent example from when I was writing the Raspberry Pi resource SLUG! (helloworld.cc/2DPHMK5). The project is a clone of the classic 'Snake' game where the player moves the slug around an 8 × 8 LED matrix, and red 'vegetables' appear at random locations for the slug to eat.

The algorithm I gave the learner to implement to randomly create vegetables and display them on the LED matrix was as follows:

- Pick an x, y random coordinate on the LED matrix
- Check if this coordinate is currently inhabited by the slug
- If it is, repeat steps 1 and 2 until you pick a location that is outside the slug
- Draw the vegetable on the LED matrix
- Store the vegetable's coordinate in a list of vegetables

When my colleague Janina tested the resource, she asked, "Why are you storing the vegetable's coordinates in a list?" I realised that I had imposed my own experience short cut on the learner without intending to, or realising I had done so.

I knew from experience that later on in the program you would need to know:

1. How many vegetables existed on the screen (to avoid having too many)
2. Whether the slug was moving into a coordinate containing a vegetable (to eat it)

...both of which would require a record of the vegetables. I was neatly sidestepping a problem before it had happened and providing a more efficient approach. However, the learner's goal, at that point in time, was simply to see the vegetables appear. Sometimes you want to allow your learners to fall into holes and write inefficient code on purpose so that they understand why they have to improve their code!

> SOMETIMES YOU WANT TO ALLOW YOUR LEARNERS TO FALL INTO HOLES AND WRITE INEFFICIENT CODE

Which approach is right?

Perhaps disappointingly, I don't have the silver bullet answer. It depends entirely on your learners – and that's why you're a teacher and your job is safe from being taken over by robots. In my opinion, you should decide how to teach concepts in the way that provides the greatest benefit to the learners you have. Are you using the turtle module with some young kids who struggle with typing? Go ahead and use `from turtle import *` so there's less for them to type. Perhaps your GCSE class is particularly good this year, so why not throw in some dictionaries and make a GUI? Is there an Oxbridge CS hopeful in your A Level class? Show them several different approaches and have a conversation about the benefits and drawbacks of each.

It's usually at this point that someone irately pipes up with "I can't believe you're condoning teaching students things that are WRONG, you're a terrible teacher". To be totally clear, I am not advising you to deliberately teach bad practice. (This is not a free pass to use `break`!) Instead I am saying it is sometimes acceptable to simplify concepts which introduce inefficiency. It's true that if you teach a programming concept in a simplified way so that a learner can understand it, you may later find it hard to persuade the learner to abandon their safety blanket and adopt a better programming practice. However, some students will never make it to 'later' if you made the starting hurdle too high for them to begin with. Which is best? Only you can decide. (HW)

HONEY, I SHRUNK THE CODE

You have a list of numbers represented as strings. You want to convert them all to integers.

```
numbers = ["23", "534",
"52", "98", "87897"]
```

A way a student might use to achieve this would be to iterate through the list of numbers, converting each to an integer in turn and adding the result to a new list.

```
method1 = []
for n in numbers:
    method1.append(int(n))
```

However, an experienced developer might use a more efficient list comprehension:

```
method2 = [ int(n) for n
in numbers ]
```

However, which method is best to teach your learners?

CHARLOTTE RUBRINGER TEACHER

FROM ICT TO COMPUTER SCIENCE

A journey from apprehension to appreciation – and some of the lessons learnt along the way

I write this as I start to emerge from a challenging phase of transition – moving from over a decade as an ICT teacher to teaching computer science instead. In the outside world, these things are too often perceived as being one and the same subject.

I love to learn as much as I enjoy teaching. As teachers, we are used to change – we have to be! Not a year goes by without amendments to the curriculum or assessments. As technology advances, we must adapt. That said, the challenges presented with the curriculum change from ICT to computer science required a huge shift in mindset and teaching style, and one of the steepest learning curves of my adult life.

My experience of the ICT curriculum was that it was largely skills-based. I spent a great deal of time looking at the backs of children's heads as they learnt to use ready-made software applications. I now understand computer science to be a much richer and more interesting subject, combining in-depth knowledge and skills with problem-solving.

Computer science, I believe, can help children to learn how to learn. That doesn't mean that I feel ICT is without value. Learning to use software and developing confidence as a user is crucial as a skill for today's world, but there is something about computer science that excites and challenges learners – and makes me a far better teacher.

Off the shelf

At the start of this journey, I was not a teacher of computer science, but a deliverer of experiences linked to coding. I would take 'off the shelf' projects (for which I am hugely grateful), and most pupils would enjoy the experiential learning opportunities that they offered. Some bright and enthusiastic pupils were able to take programming concepts and apply them to projects beyond the immediate task, but not many. My confidence was low. I felt insecure in my own subject knowledge. I was unable to solve problems, or guide pupils to solve their own problems. My attempts to talk about computer systems used regurgitated words which I had often learnt the day before. I had information, but no real understanding.

Slowly, I started to grasp enough of the coding environment and language that I was using to start supporting the pupils as they worked. I started to devise my own projects and create my own programs. At this point, I had my first experience of the sheer joy and

“ The curriculum change from ICT to computer science required a huge shift in mindset and teaching style, and one of the steepest learning curves of my adult life

SUPPORT AND RESOURCES

I am grateful to the colleagues who share their ideas and resources, and I would be lost without them.

CAS (**helloworld.cc/2uNWsaH**) has been a fabulous source of ideas and resources, both through the website and the hub meetings and training. TES (**community.tes.com**), and Facebook groups populated by generous computer science teachers have also been very helpful.

satisfaction that programming can bring. I understood why someone might punch the air, having solved a coding problem! I became one of those people, and I started to see it in some of my pupils, too. It was at this point that I started to realise I had the opportunity to teach a very special subject.

I put together a scheme of learning for Year 7, introducing Python as their first text-based language. I was nervous to see whether they could cope, especially as I had only just started getting to grips with it all myself. Each lesson had a main task, based on a programming function such as 'for loops', and each lesson also had a choice of challenges.

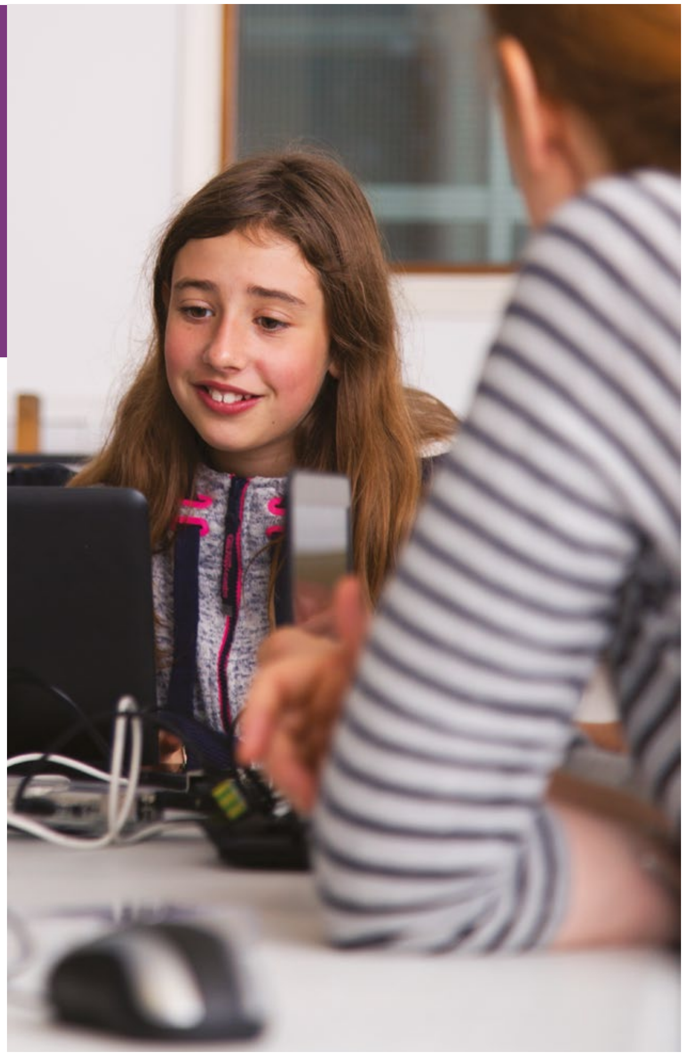
I was amazed at the response. So many pupils opted for the higher-level challenges, working collaboratively to solve the problems, and occasionally telling me where I'd gone wrong! Pupils would voluntarily ask to take home worksheets to complete the challenges.

Inspiring students

Something different was happening! This wasn't like my experiences of teaching ICT. One morning, I received an email from an 11-year-old pupil who had taken the 'if statements' from the previous day's lesson, probably stayed up far too late into the evening, and created a Pokémon-based RPG game. It worked and it was fun, and I thought, "This is why I'm here!"

I also noticed that the girls were as enthusiastic as the boys. Was this to do with my change in teaching style, or catching pupils at a younger age before some pressures of adolescence set in? I'm not sure, but I suspect a combination of the two.

Discovering the importance of unplugged activities was a real turning point. Unplugged activities make ideas much more transferable. They allow pupils to understand the functions within algorithms, and apply them in a way that is not confined to a language or a particular problem. Pupils seem much more able to take ideas and concepts away from the immediate problem when they are learning away from the machine. I'm also a fan of these activities, as they increase the level of interaction between pupils and with the teacher. I've started to see far more of their faces and hear more of their voices.



Computer science excites and challenges learners - and teachers

Training and support

Many schools struggle with funding, and time and training are two very difficult things to get hold of. I'm lucky that I had the benefit of having a CAS Master Teacher as my line manager. The school was also willing to fund the BCS Computer Science Teaching Certificate for me. I still have a long way to go. I know that computer science isn't all about coding, and my knowledge of computer systems needs work, but I now feel able to cope with what's to come.

It has been too easy for computer science to be misunderstood as being interchangeable with ICT. Yes, there is some overlap, but it is minimal. If teachers are going to be able to teach this wonderful subject effectively, those new to it need an investment of time and support. (HW)

Charlotte Rubringer is the computer science and ICT coordinator in an 11-16 comprehensive school in Gloucestershire.

EXPLAINING ARTIFICIAL INTELLIGENCE

Adding machine learning to Scratch is a simple way to introduce kids to a different approach to computing

STORY BY Dale Lane

Machine learning is an approach for getting computers to do complex tasks. Instead of describing the steps the computer should follow (as we normally think of 'programming'), you instead collect examples of the complex task being done, and use these examples to train a computer to be able to do it.

Machine learning refers to the computer's ability to learn how to do a task from examples it is given.

These systems are all around us. You use them every day.

Spam filters learn which emails we want to see, from examples of emails that have been flagged.

Translation systems learn to translate from one language to another, from examples of documents that have already been translated.

Auto-suggest on your mobile phone keyboard learns what word you might want to write next, from examples of what has been typed.

Recommendation systems for retailers, streaming music, and movie services, learning what to recommend from what has been bought and watched and rated.

Credit card systems learn to recognise fraudulent transactions, from examples of transactions.

Increasingly, the best way to get computers to do complex tasks is to collect examples to train computers to learn how to do them.

Why teach it?

Artificial Intelligence represents a major trend in computing. It's important that we sow seeds in the minds of the next generation of technologists.

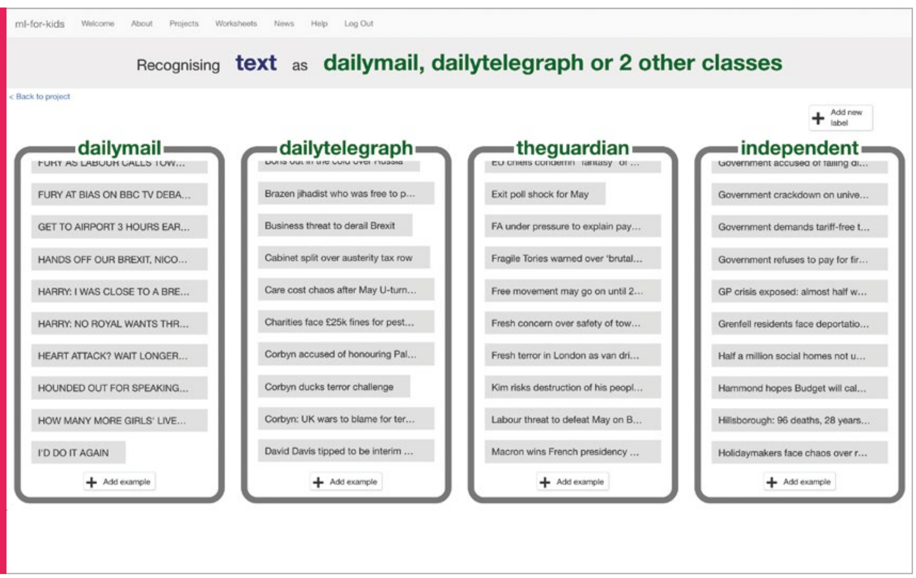
More broadly, we all strive to inform kids about how the world around them works. Artificial intelligence is now so ubiquitous, that it's hard to do that without explaining the idea behind these systems.

Looking forward, we all seek to prepare kids for the debates and decisions facing the world. An understanding of AI is essential for this. You don't have to look hard to find AI scare stories. Even if you dismiss them, some of our biggest decisions as a society will need some degree of understanding of

HOW DO THE PROJECTS WORK?

Each project comes with a worksheet that includes screenshots and step-by-step instructions, following the general pattern:

- **Step 1:** Collect examples of things you want to be able to recognise
- **Step 2:** Use the examples to train a computer to be able to recognise them
- **Step 3:** Make a game in Scratch that uses the computer's ability to recognise them



AI, whether it's about computers learning to drive cars on our roads, or computers learning to help doctors to diagnose and treat our illnesses.

Learning by making

I've helped school kids to train machine learning models to do all sorts of things. Like play the board game 'Guess Who': where they had to train the computer to understand the questions asked in the game, and to be able to recognise the pictures of the faces in the game in order to be able to answer.

We trained machine learning models to be able to play 'Rock, Paper, Scissors', where they had to train the computer to recognise what a hand looks like when it's making a rock, paper, or scissors shape.

> IT'S IMPORTANT THAT WE SOW SEEDS IN THE MINDS OF THE NEXT GENERATION OF TECHNOLOGISTS

I've helped school kids make their own version of an assistant (like Apple's Siri or Amazon's Alexa) that can understand and respond to a few simple commands. By training it themselves, they start to see how those kinds of tools work. They start to talk about how they could have trained Alexa to do the skills they use at home.

By making things they already understand well, they focus on seeing how the computer learns.

Even the youngest primary school kids I've done this with easily see for themselves how the more examples they collect, the more accurate their machine learning model will be. (In the same way that the more times they are shown something at school, the better they get at knowing how to do it.)

They quickly understand the way we test computers to see what they've learned, in the same way they are given tests and exams to see what they've learned in school, and what they still don't understand and need to be shown more examples of.

They see that if they give a lot more examples for one thing they want the computer to be able to recognise, (compared to the other things they want the computer to recognise), then it will start to favour that thing. What they can instinctively describe as needing to train a computer 'fairly' can enable a discussion about the sorts of crucial ethics issues in AI, such as training bias.

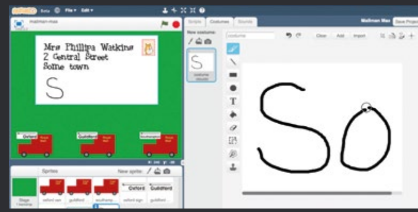
To help schools to introduce students to machine learning, I've made my resources available in 'Machine Learning for Kids' (**machinelearningforkids.co.uk**). It's a web-based tool you can use to make projects like this, and many others besides.

It gives kids a guided experience for training a variety of types of machine learning models: text classifiers, sentiment and tone analysers, image recognition systems, chatbots, recommendation engines, decision trees, and many more.

More importantly, once they've trained their machine learning model, it lets them easily make something with it. I use Scratch to do this, extending and building on the experience that both teachers and kids already have with Scratch by adding new blocks to the 'More Blocks' palette representing the machine learning models that the kids train.

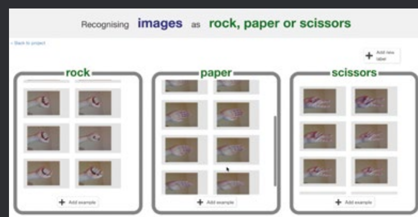
All of the games and interactive projects they already know how to make in Scratch can now be enhanced with machine learning models that they train themselves.

This combination of something new, while building on the familiar, has been tremendously effective. More importantly, it sends an important message. Machine learning isn't something to learn instead of coding. This isn't about trying to teach kids that using AI is 'better' than the coding they are learning. Rather, it's about adding this to their existing toolbox. It's about letting them know how we use AI to extend and enhance what we're able to make with coding alone. (HW)



Mailman Max
Make a postal sorting office in Scratch that can recognise handwritten postcodes on envelopes, and learn about how machine learning is used to train optical character recognition systems.

Happy Face
Learn about how machine learning is used in sentiment analysis by making a face that you train to react to the compliments by smiling, and react to insults by crying.



Rock Paper Scissors
Collect examples of what your hand looks like when making a rock, paper, or scissors shape, and use that to train the computer to play rock, paper, scissors.

Judge a Book
Train a machine learning model to recognise genres of book, based on what the cover looks like, and learn whether a computer can judge a book by its cover.

Smart Classroom
Train a smart assistant to understand a few simple commands. Use that to control virtual devices in Scratch and learn how smart assistants are made.

AGE RANGE
14-16 years

LESSON TYPE
Unplugged

REQUIREMENTS
- You will need to have covered binary, denary, hexadecimal, and ASCII conversions and binary arithmetic in previous lessons
- Pencil and paper

BINARY ESCAPE ROOM

Test your class on their base conversion skills – who will escape successfully?

STORY BY Laura Sach

Test whether your class can successfully convert between binary, denary, and hexadecimal, with the escape room challenge. Provide a series of fiendish conversion puzzles for them to solve in order to leave victorious.

Set the stage for your class

Maybe they have been captured by a mysterious witch, trapped in a cursed ancient tomb or even locked inside the deputy head's office! To escape they must solve a series of puzzles which you can make as difficult as you like. Here are some examples:

THE CHALLENGE

- Solve puzzles involving conversions between binary, denary, and hexadecimal
- Translate numbers into letters with ASCII
- Perform simple binary arithmetic

FURTHER READING

Hex words: nedbatchelder.com/text/hexwords.html

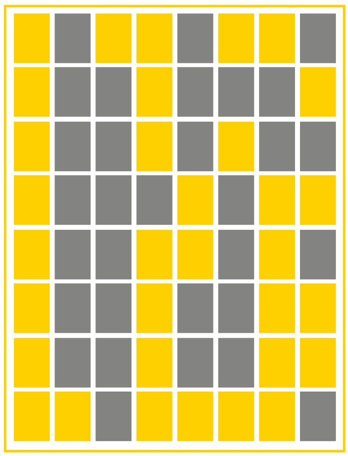
ASSESSMENT

- Show me how you worked out the answer to this puzzle
- Explain to a friend how you could create a new puzzle

Bookcase puzzle

Draw a bookcase where books are pulled out on shelves to represent binary numbers. Once they have worked out the numbers, they must translate them into their corresponding ASCII letters to work out the message.

Example diagram of the bookcase – the cells are books and the grey ones are slightly pulled out.



Solution: 'Inkwell!'

A hex

Inside the ink-well, the players find a piece of paper with mysterious numbers written on it:

Beware...this paper contains a hex!
222 202 255

When the numbers are converted into hex, they reveal a word in hex (i.e. a word using only A-F).

Solution: DE CA FF

Safe combination

Some mysterious symbols are inscribed on the bottom of the coffee cup (representing binary numbers which they have to add together to generate the safe combination!).

!!@!+!!!@!
@@!!+!@@!
@!@@@@!@+@@!!!!@
@@!@!!@@+@@!!@@@!

! = 1, @ = 0

Solution: 42 12 128 93

Inside the safe, the intrepid players discover... the key! They can now exit the escape room. (HW)

ALTERNATIVE ACTIVITY IDEAS

Ask the class to come up with their own escape room puzzles involving base conversions, and then test them out on other students. They'll have fun solving each other's puzzles, and will have to check the answers to their own puzzles too!

Explore and make

Adapt this project theme for other age groups:

5-6 years - Bee-Bots

- Create a Bee-Bot escape room – ask students to solve maths puzzles at an age-appropriate level in order to receive the directions to program a Bee-Bot to escape

14-18 years - Programming

- Ask students to program a text adventure game containing similar conversion puzzles

LEARNING IS IN THE AIR

CoderDojo's founder Bill Liao looks at the power of informal learning, beyond the school curriculum, for young programmers

STORY BY Bill Liao

April Fools' Day, and I am sitting in CoderDojo zero, the first CoderDojo in the world, in Cork. I am surrounded by parents who are drinking coffee, eating cake, and chatting. Across from us, a bunch of young people aged 9-17 are crowded around a couple of laptops, looking at the new Coding Language LOLCODE, which is actually a pretty amusing April Fool prank.

In the six years we have been running, we have seen a multitude of young people come through our Dojo, and there are many success stories. There is also a kind of magic going on. Right now I can see a young girl giving some mentoring guidance to a couple of newbies, and one of the lads has just run up excitedly to tell his dad that he has completed his new game and it's working, so he can qualify for his yellow belt, (we use coloured USB wristbands for gradings in this dojo).

There is a pleasingly industrious gentle buzz in the room and earnest faces are concentrating on difficult coding tasks. Yet there are also games being played and some decent banter going back and forth. The atmosphere is a mixture of calm enthusiasm, mingled with excitement. The dojo is a free club for young people to come and learn, (our space is provided free by the Bank of Ireland), and I might add there are some key ingredients that make the space work so well.

Good female role models make all the difference

GENDER BALANCE

We found that if you promote young female attendees to mentor the new recruits, there is a much higher degree of female attendance overall. We call this peer role modelling, and it works.

An ancient tradition

The Dojo model is over 800 years old and is used mainly for martial arts like Karate. We have adapted parts of that model for young people to learn how to program computers. We have also added on some specific features. The result is a space where learning happens, a place where you come to get really good at coding, and also a place where you can get your first taste of the magic of programming. There are over a thousand of these spaces open across the world, and in these spaces you will find a very different way to learn.

Mentoring

Instead of older teachers teaching the young, we have young mentors sharing. Instead of a fixed curriculum, we have little snippets of inspiration in the form of what we call Sushi cards. Instead of a multitude of 'don't do this' rules, we have one rule for kids: 'Be Cool'.

Instead of tests and exams, we have sharing and show and tell. Instead of desks and chairs in rows, we are in a cool startup co-working space that looks more like Google's cafeteria than a classroom. Everyone feels welcome here.

Coder-poets

What goes on here is subtle yet effective. Discipline comes from within not without. Inspiration, encouragement, and kindness infuse the atmosphere and cooperation is part of the woodwork. The goal is to allow young people to become so fluent in computer languages that they become Coder-poets, by learning their own way, creatively coding with economy of expression and art. (HW)

AGE RANGE
17-19 years

LESSON TYPE
Text-based programming

REQUIREMENTS
- Python 2.7+

OBJECT-ORIENTED PROGRAMMING WITH PYTHON

Moving from simple functions and linear programs to classes and objects with students can seem daunting, but it doesn't need to be

STORY BY Shaun Whorton

I love teaching the concepts of programming to my students. In my experience, students often find procedural programming paradigms very easy to grasp. Moving to object-orientated programming (OOP) after GCSE can be an uphill struggle for some students but, once they understand the terminology, it can open up another dimension of programming possibilities. The aim of this lesson isn't to give the full lowdown on OOP; instead it introduces the concepts of classes, objects, and instantiation.

Objects are components of a program that know how to solve certain aspects of a problem, and can interact with other parts of a program. OOP uses classes in order to solve computational problems. Think of a class as a concept and an object as an incarnation of that class. In this lesson we'll use a person as an example.

```
class Human():
    def __init__
(self,name,age,beverage):
        self.name=name
        self.age=age
        self.
beverage=beverage
```

The Human class is created. You would expect every person to have a name, age, and of course a preferred beverage, which are the properties of the class. The 'self' command is used whenever

THE CHALLENGE

- Create the Human class with appropriate objects and properties
- Create methods within the Human class and call these methods
- Create your own methods to interact with the objects

```
class Human():
    def __init__(self,name,age,beverage):
        self.name=name
        self.age=age
        self.beverage=beverage

    def talk(self):
        print("My name is {}, I like drinking {} and I am {} years old".
        format(self.name,self.beverage,self.age))

    def beverage_check(self):
        if self.beverage=='Coffee':
            print(self.name+ " is probably tired - they are drinking coffee!")
        elif self.beverage=='Tea':
            print(self.name+ " is enjoying a refreshing cup of tea")
        else:
            print(self.name+ " enjoys neither tea nor coffee. Hmmm...")

Person1=Human('Scarlett',2,'Milk')
Person2=Human('Pradeep',30,'Coffee')
Person3=Human('Elena',47,'Tea')

Person2.talk()
```

The great coffee vs tea debate! Can your students implement the beverage_check method?

we create a new object. When we create a new object, i.e. a new human, the object will use the Human blueprint, which contains these three properties. This is known as 'instantiation':

```
Person1=Human
('Scarlett',2,'Milk')
```

In this example, Person1 has been instantiated and adopts the properties of a Human. We can now access these properties:

```
print (Person1.name)
```

This will return 'Scarlett'. We can instantiate multiple objects – 'people' – by accessing the class properties again:

```
Person2=Human
('Pradeep',30,'Coffee')
Person3=Human
('Elena',47,'Tea')
```

This is one fantastic advantage of an OOP approach to software development. Once we have constructed a blueprint, we can easily create copies of objects with a single line of code. We can also create functions within our class, called 'methods', to give more functionality:

```
def talk(self):
print("My name is {},
I like drinking {} and
I am {} years old".
format(self.name,self.
beverage,self.age))
```

Note that the method takes the 'self' value: this will copy the properties of the object that is passed to it at any one time, and ensure they are available to use in the newly created method. We can call the method using the command:

```
Person2.talk()
```

Which will return:

```
My name is Pradeep, I
like drinking Coffee and
I am 30 years old
```

When we instantiated the object, we defined the properties of that object. We can use those properties throughout the methods within our class. This is a good opportunity to discuss with your students the need for reusable code, and why data integrity is important. Not all solutions lend themselves to an OOP approach, but mastering this skill could save you and your students hours of head-scratching. (HW)

ALTERNATIVE ACTIVITY IDEAS

Explore and make

Adapt this project for other age groups:

14-16 years - Unplugged

- Learners could create a structure diagram of a 'Human' and identify any methods that could be programmed from this

14-16 years - Programming

- Learners could start by creating independent functions to replicate the functionality of the methods shown in this lesson
- They could then convert their procedural design into object-orientated code

FURTHER READING

OOP concepts explained: helloworld.cc/2stZ7md

ASSESSMENT

- When is an object-oriented approach suitable?
- Describe the term 'instantiation'.
- What other methods could be useful in the 'Human' class?

AGE RANGE

7-10 years

LESSON TYPE

Visual / block-based coding

REQUIREMENTS

- Scratch 2.0
- Key spelling words

SCRATCH SPELLING TEST

This spelling game is a great way to introduce inputs, outputs, and variables, as well as learning key words for English literacy or vocabulary for a foreign language!

STORY BY Cat Lamin

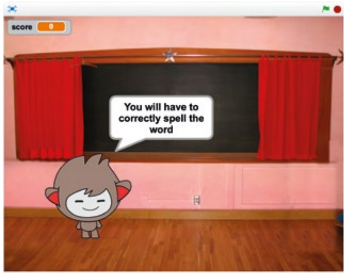
It can be a challenge to think of ways to apply Scratch and computational thinking skills to subjects outside computer science, but this is a great way to apply Scratch to the rather tedious subject of learning spellings.

This simple game teaches users about how to use Say and Ask blocks, as well as introducing variables and broadcasts, switching between costumes, and understanding conditional statements. It's based around the traditional 'look, cover, write, check' method of learning spellings that we're all familiar with, but makes it more interesting. Students could write the code to learn key spelling words for their own age group, or even make it to share with younger students. Why not ask them to plan a lesson around showing younger students how to use the game?

This game brings together a range of different coding skills, and it might be worth exploring some of them individually before challenging your students with this task.

THE CHALLENGE

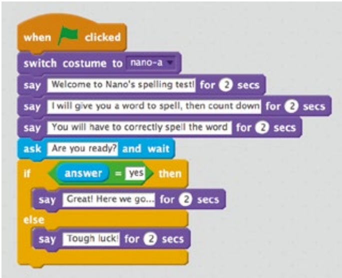
- Start a new Scratch project and choose your background
- Choose a sprite with great facial expressions; for example Nano, Pico, Giga, or Tera
- Think about how your instructions need to be written to be clear to the user
- What sort of questions could you ask at the beginning? What answers would you expect?
- If you want a countdown, you'll need to include a sprite with three costumes (the numbers 3, 2 and 1)
- Make sure you understand the difference between 'say [...]' and 'say [...] and wait for (2) secs'



The finished spelling game in action

First, build an introduction to your game using Say and Ask blocks. A Say block simply displays a message on the screen, while an Ask block expects a user response and will stay on screen until it receives one. In this case, the Ask block is looking for a yes or no answer: 'Are you ready?'. Attached to it is a Conditional block: 'if answer = 'yes' then say "Great! Here we go..." or say "Tough luck!"'. In other words, regardless of your answer, you are playing the game.

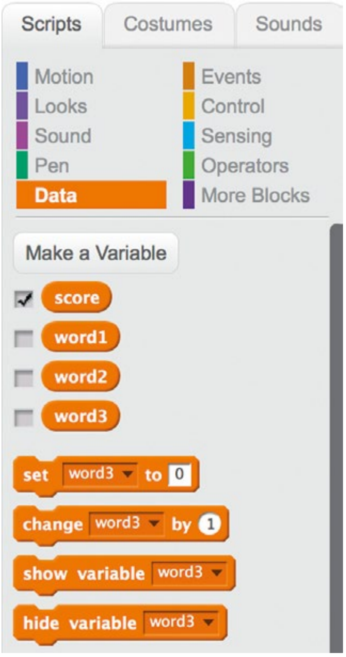
Now we need to set up our three word variables, plus a score variable. This is separate from our main code, but still included on the sprite. You can assign the variable 'word1' to



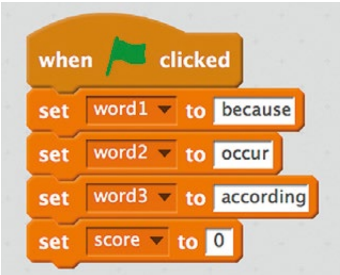
Here are the first few statements of the code. Notice that the first Ask block is searching for the answer 'yes'. You could extend the task by suggesting that the students put in an Or block to consider alternative answers such as 'yeah' or 'yup'

be any word you'd like; in this case, it's set to 'because'. You can set up the other two variables in exactly the same way. The score variable gets set to zero because we want to start the game with no points.

Now you're ready to set up your first spelling word. First, we need to tell the player what the word is by using the variable 'word1'. Then,



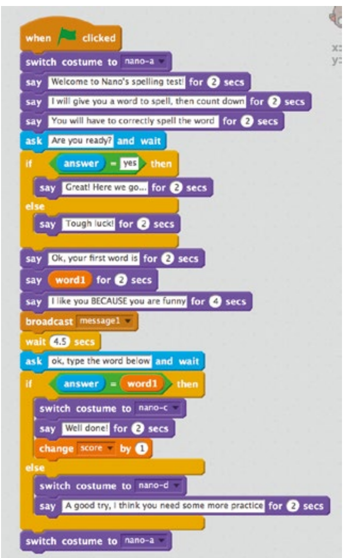
You'll need to set up three variables for words and one for score. The variable names don't matter, it's what you set them to that is important



Make sure you untick the box next to your word variables, otherwise they will appear on the screen in the game and the player will be able to copy them!

we're going to broadcast 'message1', which tells our numbers to appear and count down. When the numbers have counted down we want to ask our player to spell the word, and then use a conditional statement to check whether the spelling is correct.

Notice that we use a Broadcast block here to let the countdown know when to start. You can also see the different costume commands:

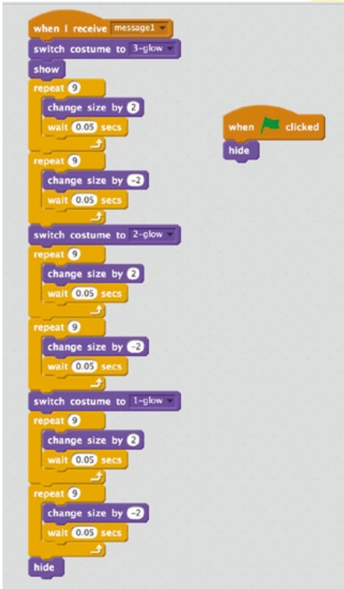


Here's the code for your first spelling word. You can repeat it with a few changes for each of your words

Nano will smile if you get the answer correct, or frown if the answer is incorrect. If the answer is correct, we have also told the game to increase the variable score by one point.

Take a look at the code you need to use for your countdown. For this project, the numbers slightly adjust in size to give the appearance of flashing.

The original version of this project had three separate sprites for the numbers, but it is much more



Using a broadcast command, you can use this code to allow your numbers to flash and count down

Make sure you add three costumes to one number instead of creating three different sprites

sensible to do this as a single sprite and switch between costumes.

Consider challenging your more able students to work out this piece of code by themselves. It doesn't actually add anything to the game, so it doesn't necessarily need to be included.

Finally, challenge students to use the score variable to get your character to announce the final score out of three. Show students how to add the score variable to a Say block, and allow them to play with their wording.

Hopefully you can allow your students to have a bit of fun with this activity. Remember to get them to peer-review each other's work, and encourage them to try out each other's games. It is really important to regularly test your code to check that it works, and this can help to spot errors.

ALTERNATIVE ACTIVITY IDEAS

7-11 years - Modern foreign languages

- Instead of using written words, consider using images and asking for words in a different language. For example, flash up a picture of a cat and ask what the word is in French
- You could develop a core bank of words to learn in a foreign language
- For a real-world application, compare this game to the early levels of Duolingo where you match images up with words!

11-13 years - Programming

- Introduce lists. Ask students to design their spelling test to randomly select a word from a list
- The sentence used as an example for each word would need to be universal to all the words
- How could you use concatenation to make your sentences work with the word selected from the list?

ASSESSMENT

- **Does your spelling test work? Does it respond in the way you expect if the answers are correct or incorrect?**
- **Could you design your game to give us the final score at the end?**
- **How would you modify your code to use different spelling words?**

FURTHER READING

Completed project: helloworld.cc/2whNqRg

Code Club ChatBot project: helloworld.cc/2uP4otO

This last bit of code allows you to calculate a final score for the player, and joins it to the phrase 'out of 3'

Think about how you could challenge your students. For example, set them the task of asking the player's name and storing their answer in a variable called 'name'. What other information could you ask for and store? (HW)

AGE RANGE

7-10 years

LESSON TYPE

Visual / block-based coding

REQUIREMENTS

- Register at **barefootcas.org.uk** to download all the resources for this activity and book a FREE Barefoot workshop for your school
- Computers running Scratch (1.4 or 2.0) for each group

THE BAREFOOT COMPUTING VICIOUS VIKINGS!

Bring history to life whilst learning about sequencing, as pupils use this Barefoot Computing resource to code an animation of a vicious Viking raid!

Barefoot Computing's workshops and resources have supported around 50 000 teachers, and reached over **1.5 million pupils.** Teachers can book a FREE Barefoot Computing workshop for their school, as well as downloading a vast range of free resources, from their site, **barefootcas.org.uk**.

The Barefoot Computing resources provide ideas for teaching computer science through cross-curricular activities. In this example, pupils make links to history as they code an animation of a Viking raid, (though of course this could be replaced with any historical event to suit your topic!). The activity develops pupils' understanding of sequencing in algorithms and code.

The lesson is in two main parts. First, pupils write the algorithm for their animation, thinking carefully about the sequence of events. Pupils then translate this algorithm into code, as they create their animations in Scratch.

The following is a summary of the full Barefoot Computing lesson plan, which can be downloaded for free, after a quick registration on the Barefoot website.

Figure 1: A screenshot from the example animation included with the resources

Vicious Vikings who?

Take the opportunity to recap pupils' knowledge of the Vikings before exploring the computing element of this lesson. Where did the Vikings come from? Why did they raid Britain? How did the Vikings fight? You might use drama for this, such as hot seating, or a mind mapping activity. See the link to the BBC Vikings website.

THE CHALLENGE

- Can pupils write a correctly sequenced algorithm?
- Can pupils correctly sequence their code in Scratch?
- Can pupils debug their code?

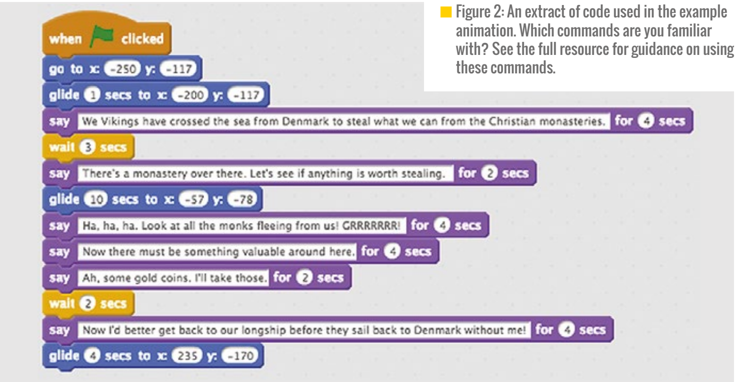


Figure 2: An extract of code used in the example animation. Which commands are you familiar with? See the full resource for guidance on using these commands.

How sequence ? Important is

Introduce the lesson's learning objective from the presentation included in the resources and, if required, spend a little time discussing the terms 'sequence' and 'program'. Open the example Scratch Viking raid animation, provided with the resource, and run this for pupils. Lead a discussion: Why was the sequence important? What needed to happen first, next, and last? Why? Would the animation have worked if the sequence were different? Why?

Let's design

Provide pupils with the animation design sheet on which they'll create the algorithm for their animation. This takes the form of a storyboard, with annotations about the sprites' movements and what they'll say.

Tinker time

It's now time to tinker! The full Barefoot Computing resource explains how a variety of 'movement', 'appearance', and 'control' commands can be used to code pupils' animations. Depending on your pupils' prior experience of Scratch, provide guidance on using these, then give pupils time to develop their understanding by tinkering with these commands.

Code, debug, code, debug

Pupils can now move on to coding the Viking raid animation they have designed. The focus throughout this activity has been sequencing, and this is still very much the case as pupils code. Have they sequenced the code correctly? Can they explain what code they have used first, next ,and last, and why? They should be encouraged to regularly run their code and debug as they go. See box 'Barefoot Computing 4 Step Debugging'. (HW)

ALTERNATIVE ACTIVITY IDEAS

This lesson can easily be adapted for different year groups, for example:

5-6 years - ScratchJr

- Pupils could use a simpler language such as ScratchJr to create their animations. ScratchJr enables pupils to draw their own backgrounds and sprites (employing IT skills), and code these to move and speak.

11-13 years - Programming

- Pupils could be encouraged to make their animations interactive, in a similar style to 'Adventure Quest' games. For example, the player might select where the Vikings raid, or where the Monks flee to! This would introduce the concept of 'selection' as means of coding these choices.

Physical Computing

- Could pupils use a Crumble, Micro:bit or Codebug to make a historical scene with parts which light up or move? A monastery with flames flickering from the roof, for example?

FURTHER READING

The full Barefoot Computing Viking Raid lesson:
barefootcas.org.uk/programme-of-study

Barefoot Computing example animation:
scratch.mit.edu/projects/25872592

Barefoot Computing on Algorithms:
barefootcas.org.uk/sample-resources/algorithms

Barefoot Computing on Sequence:
barefootcas.org.uk/barefoot-primary-computing

Barefoot Computing on Programming:
barefootcas.org.uk/barefoot-primary-computing

BBC history resources on the Vikings:
bbc.co.uk/education/topics/ztyr9j6

ASSESSMENT

- What is the sequence of events in your algorithm? Why?
- What commands have you used to code your animation? Why?
- What bugs did you find and correct? How?

SCIENCE TEACHING WITH MICRO:BIT

The BBC micro:bit is a great resource for teaching science

GUIDE BY Graham Hastings

I have piloted a number of projects with the aim of establishing a role for the micro:bit as a resource for teaching science at Key Stage 3. These projects were designed to provide a comprehensible context for the teaching of computer science, while illustrating the important role that computers play in the field of scientific discovery.

How clean is my pond?

This project is designed to fill two or three one-hour lessons, depending on the students' familiarity with the micro:bit. The objective is to create a computerised system to measure the pollution level of a garden pond, and to inform the owners when they need to turn the pond filter on.

Once the detrimental effects of excessive algae growth on a pond ecosystem have been explained, and the children clearly understand the nature of the problem, they are introduced to the apparatus. This consists of a jig supporting a bright LED and a light dependent resistor (LDR), which have been mounted so that the LED shines onto the LDR. In addition, the children are given a micro:bit, four coloured leads with a crocodile clip at each end, and access to a computer.

For the pilot, we built the jigs for the children. It would of course be possible to take a STEM approach to the project, giving the children the components to investigate, and asking them to design and make the apparatus themselves.

HOW DO I MAKE THIS HAPPEN?

Start with the people. Build a good working relationship with the Head of Science in your school, or a member of the science team who has an interest in computing. Familiarise yourself with the science curriculum. Explain to the science team why you think a project like this represents an opportunity to enhance teaching and learning in both science and computing.

The sensing apparatus

The Internet of Things is being created from billions of electronic devices connected to computers. The apparatus for this project exemplifies such a device. Although very simple in its design and function, what it does is extremely important environmentally: maintaining water quality while reducing energy consumption.

The LED providing the light source and the LDR, sensing the transmitted light, face each other, a fixed distance apart, with a space to put the water sample between them. To take a pollution reading, the pond water sample is placed in a transparent container on the wooden jig. When the sample is in place, the apparatus is covered with a box to prevent ambient light from affecting the reading. As the pollution level rises, the light transmitted to the LED reduces and the reading on the micro:bit increases.

The labelled photograph shows how the sensing components are connected to the micro:bit. The LDR needs to be connected in a potential divider configuration using a 660 Ω resistor. One end of the resistor is connected to 3 V, the other is connected to Pin 2 and one leg of the LDR. The other leg of the LDR is connected to GND.

The LED is connected by its positive leg to a 330 Ω protecting resistor. The other end of the resistor is connected to Pin 1. The negative leg of the LED is connected to GND. We used four different coloured leads to make it easier for the children to follow the wiring diagram.

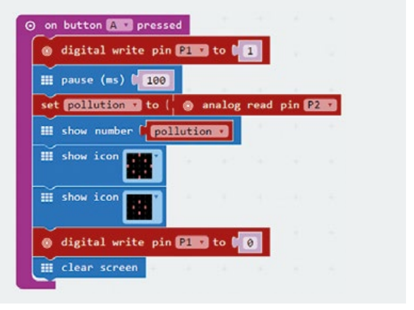


Figure 1 The calibration script

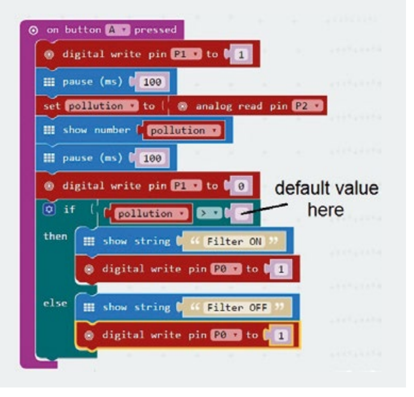
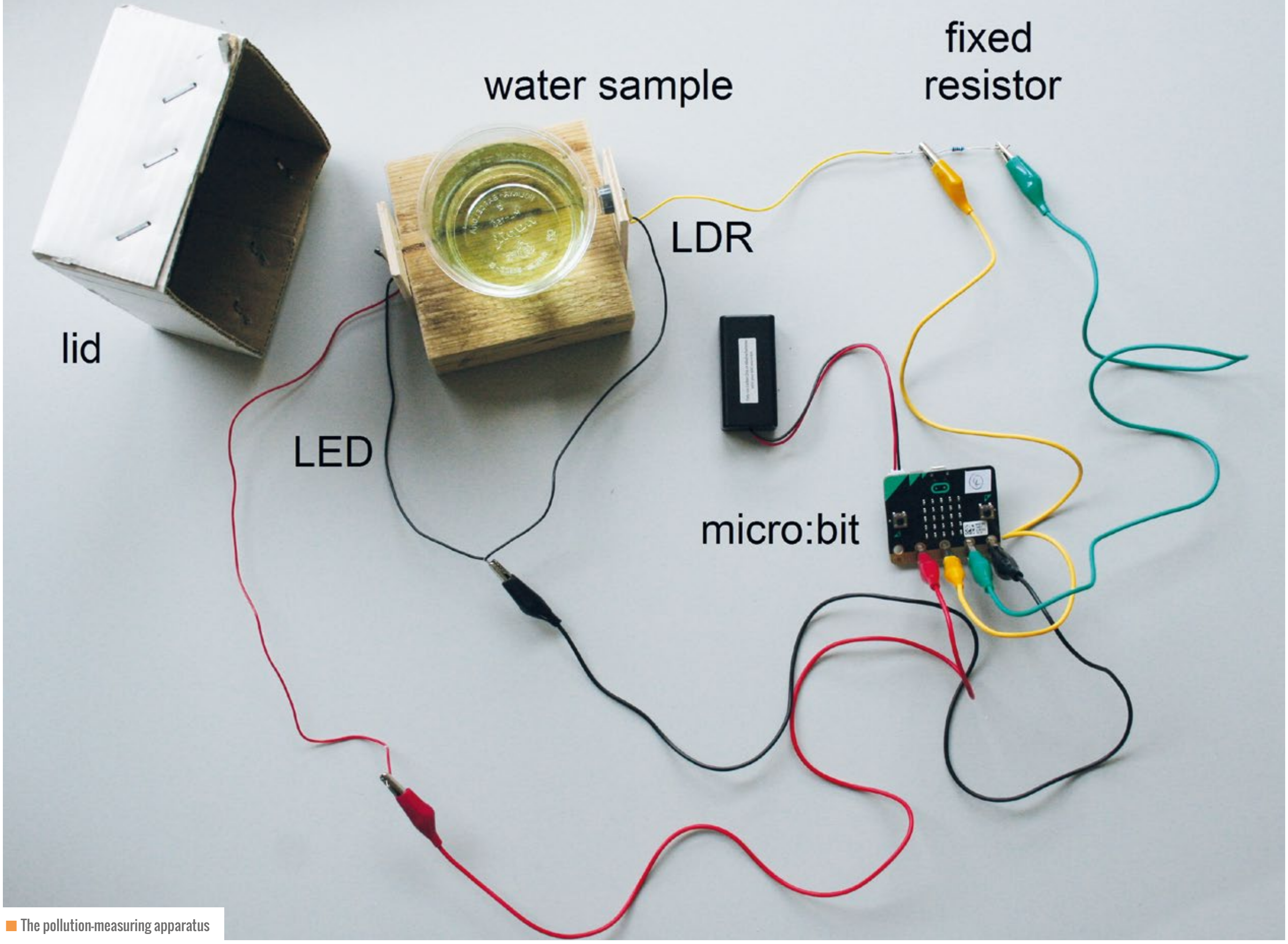


Figure 2 The complete MicroCode pond filter script



■ The pollution-measuring apparatus

Once the apparatus has been connected, the children are ready to write their programs. The Year 7 children were not sufficiently confident with Python. As they would need to program the input/output pins, I suggested that they use the Microsoft MakeCode editor (**Figure 1**).

To ensure a more reliable result, the children are instructed to take five readings and to work out the average. As an extension to their script, ask the children to edit it so that it takes the five readings, calculates the average, and then displays the result as a single value.

The pond control system

The final stage is for the children to edit their calibration script so that the pollution value is compared with the default value, and the appropriate output is displayed. This will be 'Filter ON' if the value is higher than the default, or 'Filter OFF' if it is lower.

Decomposing the problem and devising an algorithm are important aspects of this project. Some children will be able to do this, others will need a prompt. As we were short of time, I gave my Year 7s an algorithm, in the scrambled form shown, for them to resequence.

- The LED is turned off
- If the analogue value is greater than the default, display the message 'Filter ON', or else display the message 'Filter OFF'
- Start
- The analogue value of the LDR is read and this is stored in a variable
- End
- When button A is pressed, the LED lights up
- The analogue value is compared with the default value

Once they have worked out the correct sequence, the children should be able to produce a script for their pond control system (**Figure 2**). Many were able to do so. For those who found this challenging, I provided a copy of the script to help them.

This is as far as I got with my Year 7s, but there is considerable scope for extension work with this type of sensing and control system. The logical next step is to directly control the pond filter's motor, via a relay, from the micro:bit.

Detailed lesson outlines for this project and notes on many other micro:bit projects for schools are available at **helloworld.cc/2uUvfEh**. (HW)

FEEDBACK CONTROL SYSTEMS

This project could be adapted for older students studying computer science as an example of a feedback control system. It would represent an excellent opportunity to teach students about hysteresis, and the disadvantage of a single default switching value.

LORNA ELKES CAS MASTER TEACHER

USING 3D PRINTING ACROSS THE CURRICULUM

A staff member regularly enters competitions, usually successfully. This time, they won a 3D printer. Now all we had to do was get it working!

We entered this competition with a project that was already planned and underway, and we were presented with an Ultimaker 3D printer from CREATE Education at the Bett show in London.

No sooner had it arrived than it was out of its box, plugged in, and miraculously printing mini robots as a demo.

Finding cloud-based design software

We are a Google school with Chromebook technology, and consequently we use cloud-based software. Finding something compatible with Chrome and Cura (the Ultimaker 3D printing software), as well as something user-friendly and cheap, was the next challenge.

There are several options freely available, but we needed the designs to be exported as .stl files. 3D printing is a relatively new area which is still evolving, and most design tools required downloading to be stored locally. Following a conversation with someone at STEM in York, we were directed to 3D Slash (**3dslash.net**). This proved to be everything we needed: primary-friendly, and available online.

Our next challenge came with creating pupil accounts. The expectation is that users will sign up via a unique email account to access their design area and account information. As part of our safeguarding policy, primary pupils do not have individual email accounts. Fortunately, Google has a really useful fix for this. By using '+' followed by a user ID, a generic Gmail account can become multiple single user email logons for online applications.

We let the children loose.

As anticipated, they soon got the hang of it, and created some interesting initial designs. We set them the challenge of generating a 3D Windmill to match the full-sized one nearby.

Some 3D software design tools work by being 'built up' from a base that can be rotated in all directions. Sample shapes can be added or free-form lines drawn that are then filled. Alternatively, there is a 'knock down' approach, a bit like being a virtual sculptor. Generic tools are used to remove individual blocks or entire sections of the design.

Training for all

As part of the printer prize, we were offered a training session – technological know-how for staff, or a design challenge for a class of students. I cannot praise the CREATE Education team enough. They very kindly put together a bespoke training day for our children, covering technical maintenance and setup, and a hands-on design activity.

Through this training we were introduced to Beetle Blocks (**beetleblocks.com**), which uses Scratch-like programming skills to operate a logo-like bug, leaving a trail that builds up to form a 3D image. Given some mathematical perimeters, the children were soon creating their own mandalas ready for printing. You can read more about our activities on the CREATE Education blog (**helloworld.cc/2elqMxW**).

The most remarkable thing is that all this has been achieved by a teacher who is not a computing specialist, who has been able to make sense of the jargon, has not been afraid of asking for support, and has been prepared to make a mistake or two – something of which all the pupils involved were aware, and have themselves learned from. With the demands of a dynamic computing curriculum, this was by far the best part of the learning – learning for all. (HW)

Lorna Elkes is Deputy Head Teacher at Brookmead School in Buckinghamshire. She has led several subjects including D&T, Maths and Computing.

MICHAEL JONES COMPUTER SCIENCE TEACHER

AND OR NOT: GETTING IT RIGHT

Michael Jones brings hands-on Boolean logic gates to his GCSE classroom

Boolean logic can be interesting, enjoyable, and learned in a practical way. This was the promise I made to my GCSE students.

NOT getting it right

Staring at the first year of the new GCSE syllabus, I started wondering how to teach it. In particular, I was concerned about some of the drier aspects, such as logic gates.

To quote OCR,

> 'LEARNERS SHOULD HAVE STUDIED THE FOLLOWING:
> - TRUTH TABLES
> - COMBINING BOOLEAN OPERATORS USING AND, OR, AND NOT TO TWO LEVELS
> - APPLYING LOGICAL OPERATORS IN APPROPRIATE TRUTH TABLES TO SOLVE PROBLEMS'

The key word here is 'studied'. Traditionally, Boolean logic is delivered with: "This is a NOT gate. This is the truth table for the NOT gate. This is the symbolic representation of the NOT gate." Repeat for the other gates. If this works for you and your students, fine. However, for many students it is not fine, and it turns a learning opportunity into a chore.

I realised that most students could understand the theory, but found it hard to relate this to a real computer, full of circuits and electrical signals. To be honest, I didn't feel that my students were getting the full value out of this topic. With the demise in Design and Technology departments across the country, we may have lost that link between the circuit and the theory.

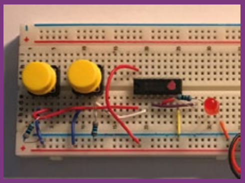
AND getting it right

What did we do about it? This year, our investigation of logic gates has been just that – an investigation. Going back to the very basics of computer circuits, armed with breadboards, resistors and TTL (Transistor-Transistor Logic) chips, we have undertaken a journey into building systems that model Boolean logic. This approach has incorporated an exploration of basic electronics, and taught me not to assume that my students understand how circuits work. In planning the sessions, I made the mistake of assuming an understanding of resistors, anodes, cathodes, and LEDs.

Working with the basic components enabled my students to embed the knowledge that a circuit is essentially the same, regardless of which logic gate they are creating. This allowed us to conduct blind testing of the chips, based on the output produced through the pressing of the two input buttons. The process was very much hands-on and unplugged: not a line of Python or Java in sight, and yet we were programming. Encourage the students to ask the basic question "why does the light go on if I hold down both buttons, but not if only one button is pressed?" From there it is a short hop to creating the truth table.

TTL LOGIC CIRCUIT

With this combination of parts, your students can build and test simple two-input Boolean systems. Just swap out the TTL to change the type of logic gate. Apart from the TTL chips, the components you need are likely to be available from your Design and Technology department cupboard. Add another chip to test the Boolean logic on two chips/gates in series.



Advanced logic

Early in the process I recognised that we had an opportunity to go beyond the confines of GCSE, and launch into A level logic gate investigations, through the development of half and full adders. Using physical circuits that were a natural development of the two-button, one-TTL chip circuit (requiring an additional input chip and some basic components to service the LEDs), we created systems that could add two bits and output the result.

As a result of their explorations, the students now understand Boolean logic. More importantly, they also understand the integral part that it plays in computer systems as diverse as calculators and aircraft control computers. (HW)

Michael Jones is a computer science teacher at Northfleet Technology College in Kent. He is a CAS Master Teacher, Raspberry Pi Certified Educator, Winston Churchill Fellow, Chartered Information Technology Professional, PGCE Subject Leader, MIT App Inventor Educator, and Specialist Leader of Education.

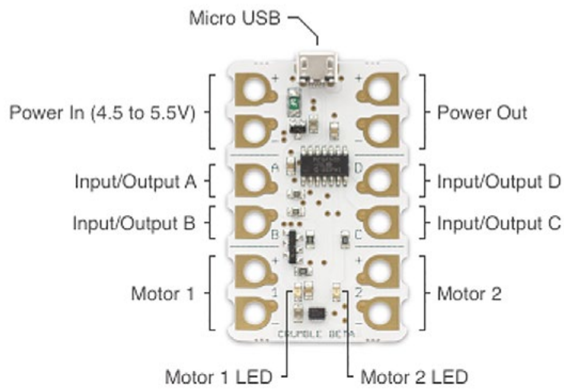
FEATURE

CRUMBLE CONTROLLER

Interested in adding physical computing to your primary curriculum? The Crumble controller is the answer! Flash lights, use switches, add motors and sensors. You can even build a self-driving robot…

STORY BY Nicholas Hughes

What is physical computing and why should we be teaching it in primary schools? In simple terms, physical computing is controlling a physical device, such as an LED or a robot, using code.



A Crumble controller

Image credit: Redfern Electronics

Wow factor

In Key Stage 2 (7 to 11-year-olds), I have been teaching programming on screen using Logo, Scratch and Kodu for a while, and I was looking for something to reignite computing with our 9 to 11-year-olds. Physical computing provides the next step for children, once they're reasonably familiar with programming – it reintroduces the 'wow' factor. I have seen many children, and teachers, give that 'Yes!' fist bump when they get their first LED to flash, or robot to move. That enthusiasm is very infectious.

So what do you need to teach physical computing? You need simple electronic components such as LEDs, motors, buzzers, and switches, and something you can program to control them. There are many pieces of equipment on the market that can do that, such as a BBC micro:bit or a Raspberry Pi, but my first choice for primary schools is the Crumble controller.

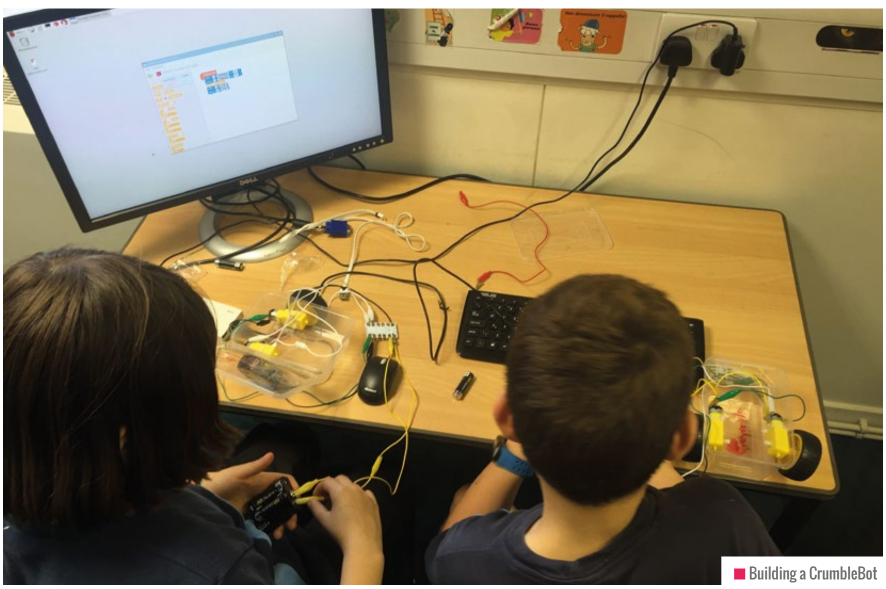
What is the Crumble?

The Crumble is a microcontroller which can be programmed to control connected devices. It is comparable to the micro:bit, CodeBug or Arduino, but it is my first choice for primary for three key reasons:

- It is very simple to program, using a block-based language similar to Scratch.
- It is relatively inexpensive – about £10 per controller.
- It is very easy to connect to a motor to get things to move. The ability to connect motors directly to the board is what, for me, elevates this microcontroller above the others.

The Crumble (see picture, above) has four inputs/outputs, connections for two motors, and connections for power. You can see that the holes are designed to allow you to easily attach components using crocodile clips.

The programming interface is very similar to that of Scratch and other block-based languages. When I first showed this to the children, they made the link to Scratch, and got straight to work because the interface was familiar. There were of course some new programming blocks that I had to



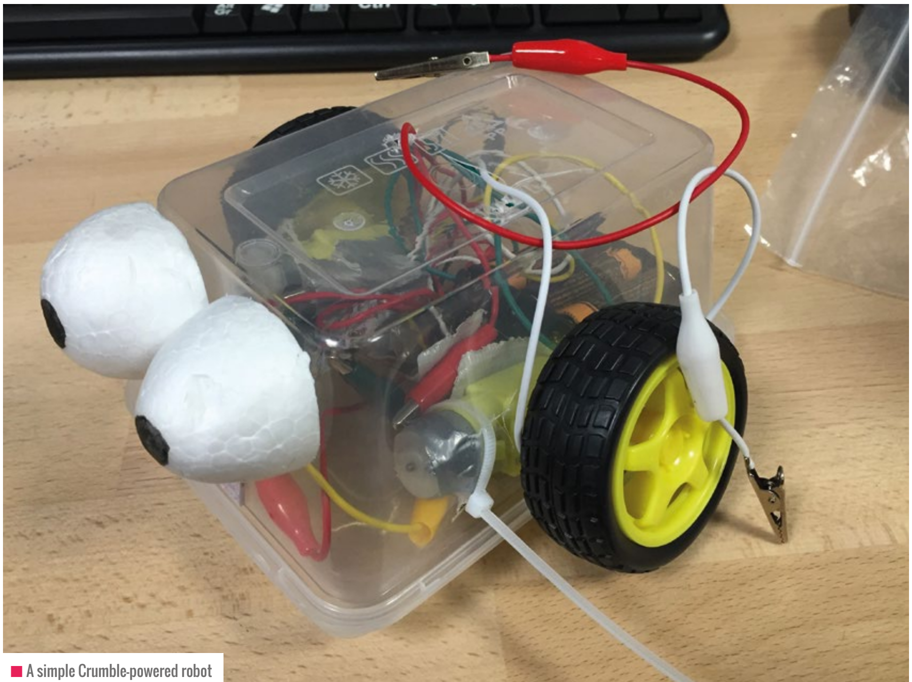
Building a CrumbleBot

explain. Something I really like about the software is that all the most common blocks are collected in one place, so this reduces the number of options you have to search through. The software will run on Windows, macOS and Linux, including Raspberry Pi.

In the classroom

So what does all this look like in the classroom? I have tried a number of routes since I first discovered the Crumble two years ago.

Currently, I start by looking at controlling a single LED and getting it to blink. Then we progress to making a simple traffic light simulation, and finally we add a switch. This progression has made sure that the children are comfortable at each step. The only thing I explicitly teach is how to connect an LED – the rest the children have to do themselves. This activity provides great opportunities for group work, as pupils discuss and plan the traffic light sequence before programming it on the computer.



A simple Crumble-powered robot

Robots

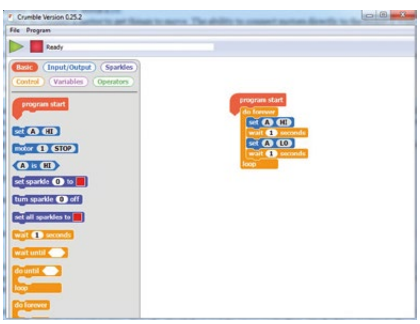
Next, I move on to making a simple robot using two connected motors. It is easy to connect motors to a Crumble. The children could design a robot chassis, or you could make one out of a simple box or takeaway container, with the two motors attached to the sides using cable ties.

When you have connected up the Crumble, can the children make the robot move forwards? It is important to start small and build up to more complex ideas. There are some great opportunities here to discuss circuits and electricity.

One of the complexities with the Crumble is that you have two elements to debug when something doesn't work as expected. If the robot spins on the spot or only one motor moves, is the error with the program or with the wiring? I like to encourage children to tinker and experiment, but sometimes I have had to focus on the programming element and help with the debugging of the wiring.

> **"THE PROGRAMMING INTERFACE IS VERY SIMILAR TO THAT OF SCRATCH AND OTHER BLOCK-BASED LANGUAGES**

Block-based programming for the Crumble



The next steps

Once the robot is moving forward, you can set bigger challenges. Can the children make it move around a maze? This can be as basic as using masking tape to mark out a course on the carpet. This is far more difficult than it seems, but a great task to try. Add a pen to the chassis, and see whether the children can make it draw a circle – or a square, a rhombus, etc.

FURTHER INFORMATION

- Redfern Electronics: **helloworld.cc/2vPhqVh**
- Crumble resources: **helloworld.cc/2eTAJlV**
- An introduction to using Crumble controllers: **helloworld.cc/2h8a5gk**

I have also added an ultrasonic sensor, so that the robot can react to its surroundings. An ultrasonic sensor works in the same way as bat echolocation. The robot will detect when something is in front of it and then react as you have programmed it. The Crumble has many sensors that can be attached to it – there is even one that will allow the robot to follow a line.

After you have made flashing lights and a Crumble bot, you could make a moving fairground ride, a waving hand, a spinning randomiser, or a reaction game. There are very strong links with design and technology and with science.

For more information visit the Redfern Electronics website (**helloworld.cc/2vPhqVh**), where you can find out about current pricing, stockists, and the latest add-ons. (HW)

PAUL POWELL CURRICULUM LEADER FOR COMPUTING

GOTO JAIL, DIRECTLY TO JAIL, DO NOT PASS GO!

Good code is not just about which statements you use

Last time round, I suggested that using lower level 'simpler' constructs to create programs might lower the bar for accessing an understanding of programming. In particular, I stuck my head above the parapet, and suggested using the GOTO statement.

Interestingly enough, most school-level educators could understand what I was getting at and were interested in the idea. University-level educators and professionals in industry were, on the whole, horrified. Having done a CS degree, worked in industry, and now teaching at secondary level, I thought it might be useful to explore the issues here.

GOTO is bad?

First and foremost, the GOTO statement should not be used in serious programs (apart from a few very specific exceptions where it is the least-worst choice.) The main reason is that it creates so-called spaghetti code. Because you can jump from any line to any other line, it means you can jump into the middle of a loop or subroutine. Understanding the structure of a program, of even just 200 lines, can become a nightmare – like trying to understand how each strand of spaghetti makes up the heap.

You can, of course, use comments and meaningful names to make the program a little bit easier to read. This is good practice with every program. The problem with this is, as the code evolves over the years, people forget to update a comment, or edit one part of the code without understanding all the rest of it. Here, the spaghetti really works against us and over the years the problem gets worse and worse.

GOTO is good?

A CS teacher, in a KS3 class in a secondary school, typically spends a huge amount of time helping students with programs of under 50 lines. Class sizes are 24 - 30+ students, with abilities ranging from learning C++ by themselves from a book in their spare time, down to some unable to work out 5+7 without using their fingers.

One way of helping students to explore code is to simplify the tools they are given. Using GOTO, instead of structured programming, means that each line of code does just one thing. It also simplifies the rules of control flow. I had the following alternatives presented to me:

```
forever:
    print “Hello”
    print “World”
```

```
start:
print “Hello”
print “World”
goto start
```

The first one is more elegant and expresses the intent. Once you understand that the indenting means that the statements belong to the forever, and that they are executed in sequence, then it can be followed.

The second one is easier to follow in terms of the correct execution sequence. I have been trying this

■ Should certain programming practices be banned?

out with a class of KS3 students for the past half term, and I am seeing a greater proportion of the lower ability students able to access and genuinely understand programs this way. When students were writing code, I used to get asked "I want the program to go back up to here. How do I do that?" Now they can play with code more easily, which helps with engagement. Next term we'll look at improving code with structured programming.

No magic bullet

While GOTOs (and other things like global variables), are often indicative of bad code, their absence does not make code automatically better. I have seen many students emulate GOTO in Python using function calls:

```
def menu():
    print("Option 2")
    print("Option 1")
    choice = input("choose an option")
    if choice == "1":
        option1()
    elif choice == "2":
        option2()
    else:
        menu()
```

GOTO IN PRACTICE

Most modern languages do not offer a full unrestricted GOTO. Some languages, such as Python, don't have a GOTO at all. Others, such as C, allow GOTO, but only within a function. Professional coders review each other's code and, using a GOTO, would certainly raise eyebrows.

GOTO is never absolutely necessary. In a few situations (such as breaking out of multiple nested loops), it can offer the most elegant solution. In some cases, GOTO is also the most efficient solution. Have a read around and make up your own mind.

Coding style is something that is developed over years, and often the best learning opportunities are when a student gets themselves in a right old mess. Many of these problems only really exhibit themselves when we start to write larger, more complex programs. I don't think GOTO is quite the bogeyman it has been made out to be. I can see it working as an educational scaffold in my classroom. There are legitimate uses in industry – but just like me, be prepared to justify yourself to your fellow programmers! (HW)

Paul Powell is Curriculum Leader for Computing at George Mitchell School in East London. He also co-leads the Waltham Forest Hub.

PRIMM: SOMETHING FOR YOUR PROGRAMMING PEDAGOGY TOOLKIT?

Do your students find programming difficult? Perhaps try PRIMM!

STORY BY Sue Sentance

Do your students find programming difficult? Are they staring blankly at a blank IDLE window not knowing where to start? Well, they are not alone. Research in higher education has shown that many students find it difficult to write a program from scratch – according to 'Designing for deeper learning in a blended computer science course for middle school students' by Grover, Pea and Cooper, from Computer Science Education, Vol 25, No 2. As teachers, you probably use a variety of techniques to help them, presumably some of the strategies that helped you yourself to learn to program.

The PRIMM method is one approach that might help you to teach programming and add to your toolkit of useful strategies. It gives some structure and a memorable way to think about teaching programming that you may find useful. It is independent of language, and can be used for primary and secondary-aged children.

What is PRIMM?

PRIMM stands for Predict, Run, Investigate, Modify and Make. The Predict phase involves students looking at a short program and trying to decide what it will do. This can be a starter, can be done in

■ PRIMM is one approach to teaching programming: Predict, Run, Investigate, Modify, and Make. Try it and see if it helps!

EXAMPLE STARTER CODE TO PREDICT

What does this program do? In your pairs, draw on your whiteboard what you think the output might be when the code runs.

```
from turtle import *

def triangle():
    forward(150)
    right(120)
    forward(150)
    right(120)
    forward(150)
    right(120)

triangle()
penup()
backward(200)
pendown()
triangle()
```

pairs, or you could even spend a whole lesson on this phase.

The Run phase involves downloading the code, and then running it to check your prediction. It does not involve any copying in of code (not my favourite strategy for children learning programming) and, as the program is written by the teacher, the student can focus on what it does and not whether they typed it in correctly. The third stage is Investigate, which should be varied – there are many, many lovely exercises that you can do to get into the nitty-gritty of the code – annotate it, use Parsons puzzles to get it in the right order, put errors into it and do some debugging, trace through it (use the PLAN C TRACS activities for this), label the variables, etc. The fourth stage is to modify the code by changing first something simple, and then make more and more modifications, which can add lots of differentiation to your class and finally, make a brand new program, when you can borrow bits of code from the original program, but it will have a new function, context, or problem to be solved. These five phases may not be in every lesson, depending on the topic. And you may cycle through the first three or four phases several times.

PRIMM is based on research into programming. In particular it adapts an approach called Use-Modify-Create (UMC) which was proposed in 2011 by Irene Lee and colleagues. It fleshes out UMC, and also emphasises the importance of reading and understanding code, which has been shown by many researchers to be a key indicator of whether you can write programs, and also differentiates between the structure and the function of code, following some work by Carsten Schulte and colleagues on the Block Model.

Does it work?

We cannot yet demonstrate evidence that PRIMM works, there's still some experimentation to be carried out, of course, but it has to be said that our first trials in a small number of schools do already seem to show quite clearly that teachers, for a start, have found it really useful, and even more importantly, that students show an increased understanding of the key concepts of programming thanks to PRIMM.

One teacher said, "It was amazing! In one lesson, they pretty much all got the concept of a function".

In addition, Phil Bagge, a well-known primary computing expert, has recently developed some new resources that were inspired by this approach. We ourselves use this approach in running our KS3 Computing and GCSE Computer Science evening classes.

Our research is continuing and we are looking for teachers who may be interested in getting involved in a second pilot study. Check out the Kings College London Computer Science Education Research blog for more details.

Happy PRIMM-ing! (HW)

EXAMPLE QUESTIONS FOR INVESTIGATE AND MODIFY

Code to predict and run

```
def shopping():
    my_list = []
    next_item = input(“What do you want to buy? “)
    my_list.append(next_item)
    for each in my_list:
        print(each)

shopping()
```

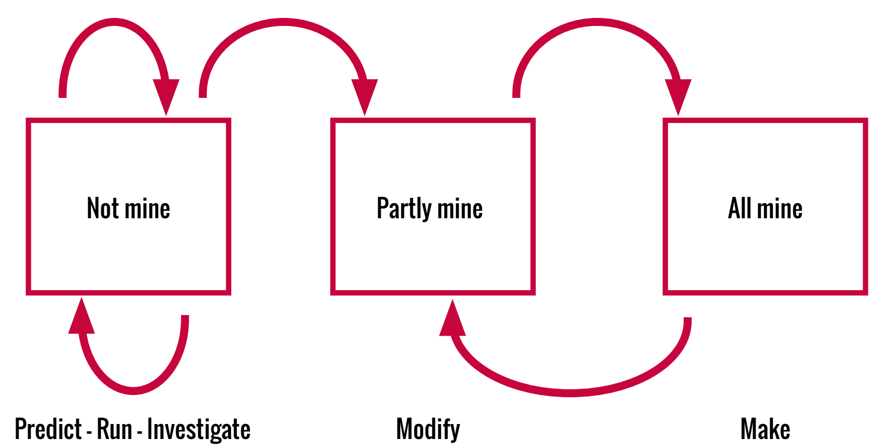
Example Investigation questions

01. What does [] mean?
02. What would happen if the line my_list = [] was deleted?
03. What is the variable next_item used for?
04. Why is next_item not a good name?
05. What does the for loop do?

Example Modify activities

01. Change the program so that the user can add 5 items of shopping
02. Change it so that the user can add as many items as they like and type "999" to stop. This is called a "rogue" value.
03. Using either version from 1 or 2, check before adding to the list that the item is not already in the list.

■ By running the teacher's code first the ownership of the original program is with the teacher – and gradually passes to the learner. This helps confidence



PINARY WARS: FROM PAPER TO THE RASPBERRY PI

Creating resources that promote learning is an exciting process. We did this, and increased the excitement at Picademy by making it for the Raspberry Pi too!

STORY BY Ramzi Ramzi

The task given to me during my PGCE year was to create a resource that would help KS4 students, and below, with their binary addition. The class I was teaching at the time enjoyed some friendly competition between each other. With these two facts in mind, a game called Binary Wars was born.

The game involves four card cut-outs: one side showing a '0', the other a '1', each representing a bit. A student would use these four cards to create denary values ranging from 1 to 15, which they would draw randomly from a pile next to them. A mystery 'Binary Crash!' card was also in the pile, but we'll save that for a bit later! Two students would face off against each other, with the first one finishing their pile declared the winners.

To stop myself from becoming dizzy, making sure each student created the correct binary representation of the denary, I assigned a student judge for each player. Each judge had an answer sheet showing all binary values a player would create. This not only enabled me to commentate on the furious binary calculations occurring, but also students finding the task more difficult had a chance to participate in the game. To ensure the judges were always on their toes, the 'Binary Crash!' card was introduced. When a player draws this, the judge can decide what number the player must produce. This gives some ownership and interaction to the judge, as well as the unintended side effect of seeing how much they like their player depending on the difficulty of the number chosen.

Binary Wars had a great reception with fellow PGCE teachers and tutors alike. Whilst their binary conversion skills increased, so did the volume in the room as the players came closer and closer to finishing. As passionate screams of joy filled in the room, I wondered how to improve upon the task. A Raspberry Pi Picademy event was also fast approaching. Enter Alex Parry, and the joys of the Raspberry Pi.

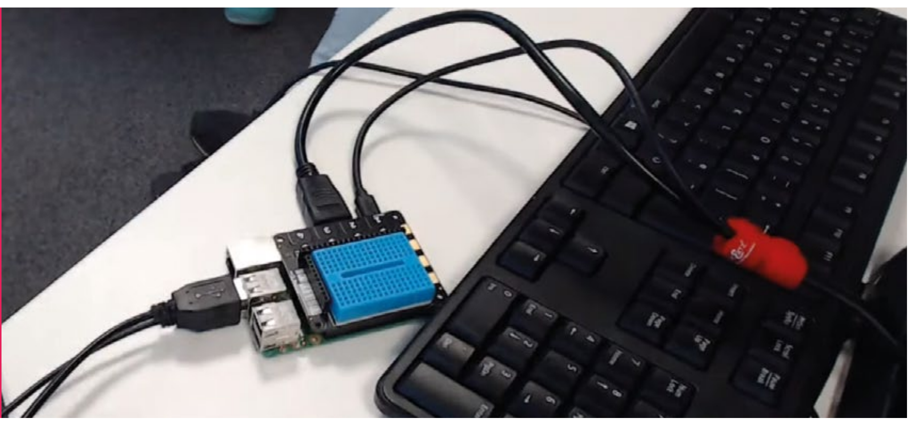
The training

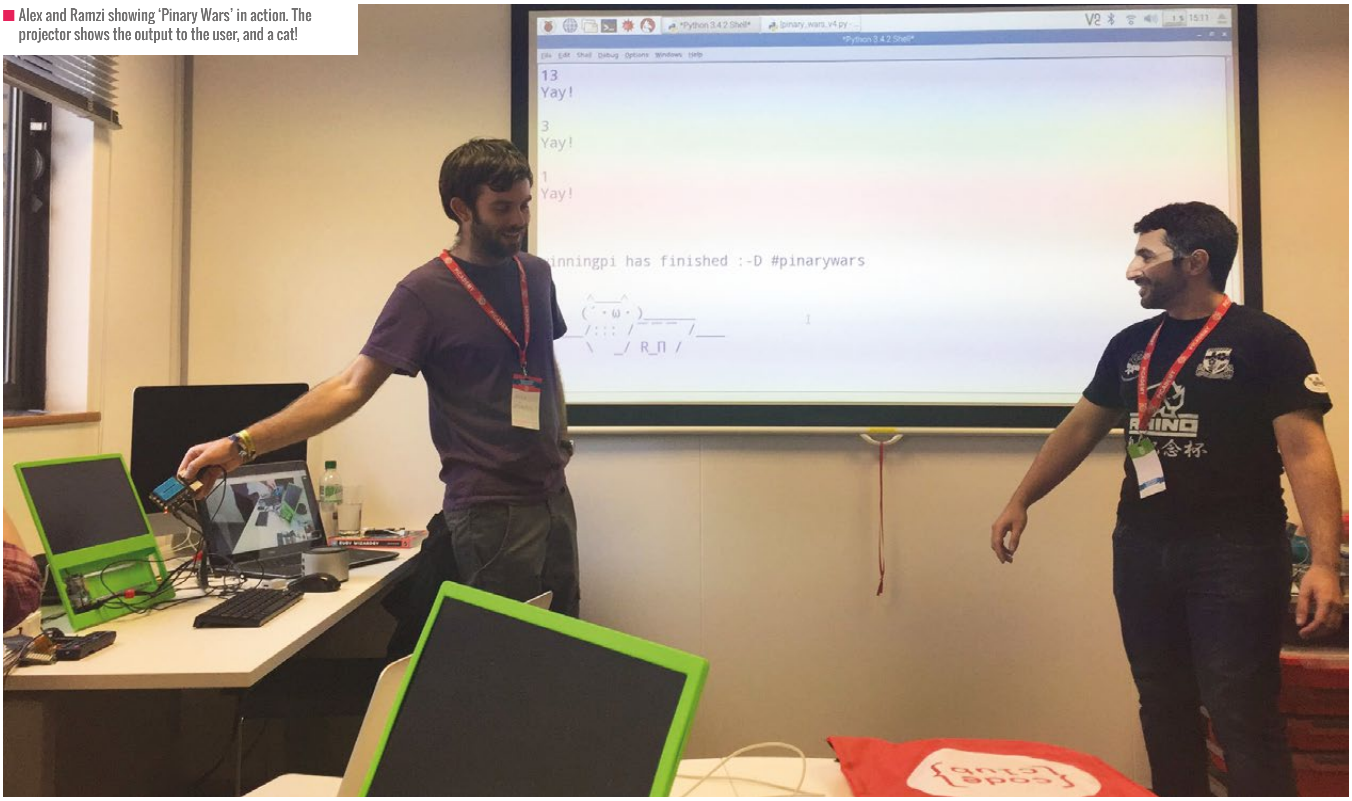
I was lucky enough to attend a Picademy CPD event at The Raspberry Pi Foundation in Cambridge. As educators, we encourage our students to pair programs, and highlight the many benefits in doing so. In order to see these benefits first-hand, I teamed up with an incredibly talented programmer and fellow PGCE student, Alex Parry. After the first run of the card-based Binary Wars with our PGCE peers, we began talking and thinking of ideas to bring this to a digital medium. Shortly after realising we also had a whole day at the Picademy to create our own project, we decided the Raspberry Pi would be an ideal device to try it on. The question being, how?

During the first day of Picademy, we got to experience many different ways the

MORE THAN FOUR BUTTONS

Here is the Raspberry Pi, with an Explorer Hat on top, with our program running on it. We only utilised one small part of what the Explorer Hat is actually capable of. Some of these included crocodile clip pads, a mini breadboard, coloured LEDs, and its own Python library to explore!





Alex and Ramzi showing 'Pinary Wars' in action. The projector shows the output to the user, and a cat!

device can be used. From various sensors surveying the world around us, to using Python to manipulate Minecraft on a deeper level, the Pi has it all. Many 'hats' can be put on top of the Raspberry Pi to enable new functionality. During the first day of training, the Explorer Hat caught our eye. One of its sides has four touch-sensitive buttons, each with an LED. Binary Wars fundamentally worked around four bits. I sense a match!

After an incredible first day of learning what the Raspberry Pi was capable of, we began work on the newly and creatively named 'Pinary Wars'. The core of it was coded in Python. Our pair programming experience turned out to be a mixture of the more traditional 'driver' and 'passenger' method, and going into our respective corners to tackle particular problems. At times, I may have been working on the logic behind the button pushes and getting the correct answer, with Alex working on the structure of the program and generation of questions. Working in a pair – quite fittingly – definitely improved the overall quality of what was made, and of course sped up the time it took to create anything.

Now to unleash the results of our toil to everyone at the Picademy!

> IF ANYTHING, THIS PROCESS HAS SHOWED ME THE LEARNING POTENTIAL IN THIS JOY, AMPLIFIED BY CREATING IT WITH A FRIEND

The finished product

We ended up with two Raspberry Pis running 'Pinary Wars', using the Explorer Hat for input. Users would be presented with randomly generated denary values up to 15, using the four inputs as four bits on the Explorer Hat to input its binary representation. We then put numerous players on each device, each responsible for one or more of the inputs.

The development process from our side as educators was incredibly enjoyable. Computer Science is a field that often encourages its learners not only to become efficient users of technology, but to find the joy in creating and unleashing those creations to those around them. If anything, this process has showed me the learning potential in this joy, amplified by creating it with a friend and peer.

There may be many other enhancements on this idea that you may have come up with just reading this. What we plan to do next is use the Raspberry Pi camera and an API to recognise faces. Using this facial recognition as input, students could stand up for a '1' bit, and sit down for a '0' bit in order to create denary numbers. Learning and exercise, all from binary! The possibilities are endless.

The presentations were streamed live, and can be found on the Raspberry Pi official YouTube channel ('Picademy Cambridge – February 2017', with Pinary Wars featuring around the 59m 50s mark). Follow our twitters, **@RamziSensei** and **@ParityParry** for advancements on Pinary Wars, and our other creative endeavours.

We would like to thank King's College London, and the Raspberry Pi Academy, for giving us the opportunity and platform to bring this idea to life. (HW)

FEATURE

The discussion was warm and lively, lots of learning

ALAN O'DONOHOE AND HIS "FLIPPIN' CLASSROOM"

What is best practice with flipped learning, and how can I use it in my classroom effectively to support learning and not create mountains of extra work for myself? Thoughts from Leicester Secondary CAS HUB, 2 February 2017, & Nottingham Trent Computer Science PGCE event

STORY BY David Chaplain


Alan attunes his quizzes to individual skills

Most teachers are probably aware of the concept and rationale behind the flipped learning approach, as pioneered by Alan November – the basic principles of a flipped classroom are to deliver content outside of the class and to move active learning into the classroom. However, being aware of the pedagogy, and putting it into practice in your own classroom, are two very different things indeed. It can be fraught with issues; "how do I resource it"? "What becomes of my role as teacher"? "What to do if learners don't do what is expected outside of lessons"? "How will observers (Ofsted, SLT) see progress"? Alan answered these, and demonstrated and discussed with the audience how he used this approach to very positive effect in his own classroom, through demonstrating the use of his MOOC, (which is free to use for the first twelve units), with its index of relevant written and curated video materials, (so as to avoid the usual distractions of skateboarding canaries often found on YouTube), randomised tests, and tracking documents.

SOME COMMENTS FROM TEACHERS AND PGCE STUDENTS

It was interesting to see how it could be approached and the idea of the students producing an image was brilliant! (Teacher)

Brilliant session! I have so many ideas now to take away and start teaching the new GCSE computer science course. (Teacher)

The class I have are so bright but the girls are weak on code, so I think this approach will help them. (PGCE student)

I will be using it and trying out flipped learning without being scared because Alan presented it, in such a way, that it seems productive to students and teachers. Not "just go away watch this video and we will talk about it". But there are pre and post-tests, there's the recording of information in fun ways, there's coding, and it's not just theory! (Teacher)

Wow! The guy is amazing coming from a non-specialist Computer science teacher! His session was fun and he's a really cool guy! (PGCE student)

Thanks for arranging this. The best CPD session I have had :) (Teacher)

Nobody left the event without exciting new ideas

finding similar things for them to do back inside the classroom". Alan very carefully described, with very practical examples, how he dealt with those students who didn't "buy into the model" and hadn't done the out-of-lesson work – how these were whittled down to a hard core of one learner in his case. He also described how he used a simple drawing and labelling technique, (simply peer-marked out of 5), to help his students remember and record what they had done in "flipped mode". Finally, he discussed how he managed his classroom once the students were there. This was through active group tasks and

> I HOPE TO INVITE ALAN TO COME BACK IN THE NEAR FUTURE TO LEAD A SESSION ON DEVELOPING PEDAGOGY

Flipping the classroom - best (and easiest) practice

However, the real strength to Alan's discussion was his very charming and witty stories of how he approached flipping in his classroom, using in-class activities that didn't simply, "replace the teacher with an electronic resource and leaving the student to get on with it themselves outside of school, then flexible quizzes aimed at individual learners' progress, rather than the more common "today we are all going to learn... (whether you already know this or not)".

Both sessions ended after two hours, with PGCE students and teachers all positively discussing teaching and learning in Computer Science which, if nothing else, is a good thing and doesn't happen enough. I hope to invite Alan to come back in the near future to lead a session on developing pedagogy around "pair programming". Finally, I'd like to thank Alan for his time and ideas from the Leicester CAS hub and from the trainees at Nottingham Trent Computer Science PGCE course. (HW)

SOME STARTING POINTS

- Alan's MOOC
  courses.exa.foundation
- EAX foundation (courses)
  exa.foundation
- Alan's schedule
  helloworld.cc/2Fpnbhg

SOME FURTHER READING

- Classroom strategies for CS education ML Maher et al. (2015)
  helloworld.cc/2CXdzM5
- Flipped Learning: a response to five common criticisms (2017)
  novemberlearning.com

SAMANTHA BALORO RASPBERRY PI FOUNDATION RESEARCH INTERN

DISCOVERING COLLABORATIVE PROBLEM SOLVING

Could the future of learning be working together?

Instead of teaching in routine ways, imagine basing lessons on open-ended questions such as 'how much paint is needed to paint a classroom?', with the aim of developing collaborative skills and problem-solving. Collaborative problem-solving (CPS) is loosely defined as the ability of a group of people to work together on a shared problem. A recent report commissioned by Nesta, and written by academics at UCL, argues for a greater use of CPS in education.

Collaborative problem-solving brings together individual problem-solving and the process of collaboration, and is one of the most important skills needed for this generation. However, it can be very easy for collaboration to become a one-sided affair, with one person taking over. The 'success' of the collaborative activity depends on the skills and attitudes of learners in relation to each other, as well as the type of activity.

Slavin (2015) states that the essential features of collaborative learning are:

- **Positive interdependence**
  Learners work harmoniously together without one member taking over
- **Promotive interaction**
  Learners support each other when completing tasks
- **Individual accountability**
  Making sure that each learner is committed to their section of work, and feels responsible for the group's task success
- **Interpersonal and group skills**
- **Group processing**
  Learners reflect on their working relationship, and consider how it can be improved jointly and individually.

Studies have found that engaging in collaborative group-based learning promotes positive attitudes to schooling, as well as improving the social climate of classrooms. It also results in learners being more actively engaged in their learning and having higher levels of motivation.

The reason for this change in students' attitudes to learning may be because engaging in CPS involves mechanisms such as students being able to:

1. Articulate, clarify and explaining their thinking
2. Listen to ideas from others, which in turn leads to developing understandings in areas which were previously unclear to them
3. Resolve conflicts by presenting counter explanations, evidence and arguments to others as well as actively engaging in the construction of ideas and thinking in order co-construct knowledge.

Current research in UK schools has found that, although learners are often seated in pairs/groups during school activities, it is rare that active collaboration occurs in ways that are cognitively beneficial. When examining the amount of collaboration occurring in different subjects, it was also found that maths and humanities students were less likely to participate in collaborative work than science students.

The reason for the scarcity of meaningful collaboration problem-solving occurring might be due to a number of barriers. Here's a shortlist of potential issues considered in the research:

- The gap between CPS and the current National Curriculum, which focuses on exams
- High workloads teachers are faced with
- Teachers being hesitant to practise CPS because they will have less control over learners
- Teachers not having enough training and confidence to teach CPS
- Students not enjoying working in groups

The success of CPS also depends on how teachers organise, engage and set up tasks as well as support groups. This is very often difficult to get right as nobody can just assume that by putting people in groups, collaboration is going to naturally occur. It is important to avoid being too directive as it can negatively affect group collaboration. When facilitating CPS, teachers should monitor group interactions while asking open ended questions to challenge students and encourage them to reflect on their views.

Behaviour management is another aspect which needs to be considered. CPS can lead to increased noise and disagreements amongst learners. It is important to gradually and frequently introduce CPS activities in the classroom so that learners and teachers can gradually develop the skills needed for productive CPS such as self control, productive argument etc. Most importantly, it is not possible to implement collaborative problem solving without the active support of a school's senior leadership team. This involves ensuring that they understand the importance of CPS and allow teachers time to attend training and embed it into practise. In short, it's not an issue which can easily be dissolved.

> “ Teachers can gradually develop the skills needed for productive CPS such as self control, productive argument etc

But it's worth bearing in mind that research shows that when done right, CPS can singnificantly benefit children's cognitive attitude to learning, as well as be a considerable aid when it comes to their development of crucial skills such as teamwork and problem solving. However, implementing it involves a tricky balancing act, and more research is needed on collaborative problem solving and it's application in the classroom.

The full report is available on **helloworld.cc/2CKmaSa**. Nesta have also completed an exploratory pilot looking at collaborative problem solving discussion in the classroom, available at **helloworld.cc/2m3bGns.** (HW)

Samantha Baloro is a psychology graduate and researcher. She completed the Raspberry Pi Foundation Research Internship in 2017.

SHARED CODING, TINKERING AND OTHER TECHNIQUES FOR TEACHING PROGRAMMING

What strategies and techniques do you use to teach programming? Have you ever audited your lesson planning to see how much scaffolding you provide children with as they learn to program?

STORY BY Jane Waite

In primary, whatever lessons we are teaching we use approaches to 'deliver' material. For example, we might demonstrate how to solve a problem in maths. In literacy, we might give children targeted activities, such as answering comprehension questions, fixing sentences with mistakes, comparing and evaluating text. In D&T lessons, we might set up lessons where children explore how examples or parts of a product work before designing a new item. In PE, we might ask children to adapt a gym routine.

Have you considered what techniques you use when teaching programming? Or considered the balance of approaches used, and which of these options is most effective for teaching different groups of children for learning different concepts and skills?

Techniques to consider

Here are some ideas of techniques and approaches that we might be able to use when teaching programming:

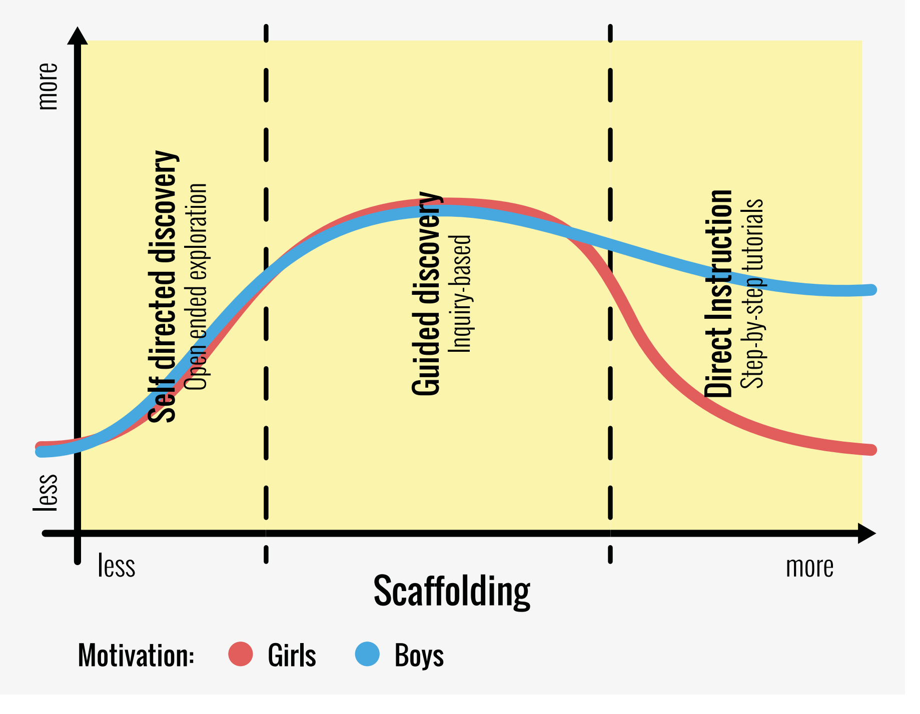
- **Copy code.** This technique is where pupils are given a set of instructions to follow to create a program. The program was thought of, designed, and coded by someone else, pupils are now re-creating it. Learners are required to follow a set of instructions, line by line. This might be through an online teaching product or might be a printed set of instructions.

- **Targeted tasks**. There are many targeted tasks that teach specific concepts or address particular misconceptions. Often

PUPIL MOTIVATION

Emerging research evidence indicates teaching strategies may have an impact on pupil motivation.

This diagram from 'Scalable Game Design: A Strategy to Bring Systemic Computer Science Education to Schools through Game Design Creation' (Repenning et al. 2015) implies that girls are less motivated by a copy code approach than boys and that both boys and girls are less motivated by a purely tinkering approach. But further research is needed to verify this work.



Copy code	Targeted tasks	Shared programming	Guided exploration	Projects (idea/design/code/debug/share)	Tinkering
				• Use • Innovate • Remix • New	

■ What approaches and techniques do you use to teach programming?

such tasks are aimed at getting children to read and understand code. If learners are online, there is a the temptation to run code before you have read it, so these activities may use printed code snippets. An example activity might be to give learners a program and ask them to summarise what the code will do, or to trace the code line by line. Simply put, tracing is where you say exactly what each command will make happen when it runs. In both of these scenarios, learners are predicting what the code will do. Other targeted tasks might be to spot the difference between code snippets, remix code to achieve a particular outcome, and fixing buggy code.

■ **Shared programming.** This method is very similar to shared writing, where the teacher knows what they want to teach and is showing pupils, not only what the finished product looks like, but is also explaining the making process and their thought process. It could be that teachers or pupils deliver the demonstration. It could be that the teacher takes ideas from pupils as they are demonstrating to create a 'class' version of the program. This form of apprenticeship might also be used in small groups or on a 1:1 basis.

■ **Guided exploration.** Here learners are provided with just two or three commands that they must explore. The teacher has an idea in mind of what they want children to learn. But rather than telling them what the commands do, they ask children to find out. Teachers might include some questions to nudge children along to get to the objective of the task.

■ **Projects (use/imitate/remix/new/share).** Here pupils are required to create a project of some kind. There are lots of different ways to run projects, as well as different ways to scaffold learning in projects. In literacy, some teachers follow a progression that scaffolds learning to write texts. At first, pupils read lots of examples of the genre of text they are going to create. Then they create an imitation of an example text. Next, they create a variation of the text. Finally, they get to inventing a brand new version. In programming projects we could do the same, start with children using example projects. Next, creating a project that imitates a high quality exemplar. Then, have projects where we are remixing ideas, algorithms and code examples; with an end goal of learners independently creating a brand new program. This development of independence might span years of different projects across different programming genres.

■ **Tinkering.** This techniques requires pupils to play. They are given access to a programming environment, and perhaps some hardware too, and asked to explore and play.

This list of approaches is by no means exhaustive and the techniques are not distinct. When running a project, you might include a little copy code to get children to accomplish one particular tricky thing, that you plan to later teach, through guided exploration or targeted tasks. Or you might develop children's curiosity and emerging understanding of a new programming language with some tinkering, before demonstrating certain features and moving on to a 'remix' project.

> “ YOU MIGHT INCLUDE A LITTLE COPY CODE TO GET CHILDREN TO ACCOMPLISH ONE PARTICULAR TRICKY THING

Further ideas

Whether you use a 'shop-bought' scheme of work or have created your own, you might find it interesting to audit your planning to see which of these approaches are included. If your class is not developing independence in programming projects, perhaps reflect on the techniques you are using and try a different approach. Or one of these approaches might be useful for an intervention or for differentiation.

Here, we have not commented on the impact nor effectiveness of the approaches outlined. There is limited research to tell us which techniques are best for primary pupils. Each of us probably has views and personal experience on each method. Why not write an article for Hello World on your experience of using different approaches? (HW)

MARK THORNBER TEACHER

MATHEMATICAL MUSINGS

Finding a handy short cut to calculate big powers

Last time, I looked briefly at the workings of the RSA algorithm for internet encryption. At the end were a number of questions, including 'How can we calculate large powers very quickly?' An example might be 244^{321} (mod 1000). I've chosen modulo 1000 here to make it easy to follow for humans because we like to work in base 10, and now we only need to keep track of the last three digits as we work. Computers are happy to work in other number bases quickly too.

Imagine you had to work out 244^{321} using only pencil and paper. It's a daunting task! It looks like you have to do 321 long multiplications – there must be a short cut!

In fact there is a shorter method, based on properties of powers. Suppose we want 244^8 (mod 1000). We can get this with three multiplications instead of eight:

- Work out $244^2 = 244 \times 244 = 59\,536$, so $244^4 = 536$ (mod 1000)
- Now, $244^4 = 244^2 \times 244^2 = 536 \times 536 = 287\,296$, so $244^4 = 296$ (mod 1000)
- Finally, $244^8 = 244^4 \times 244^4 = 296 \times 296 = 87\,616$, so $244^8 = 616$ (mod 1000)

This method will give all powers that are themselves powers of 2, i.e. 1, 2, 4, 8, 16, ... Fortunately, these form the basis of binary numbers, so we can make all other powers from these ones. In our example, 321 = 256 + 64 + 1. We need eight multiplications as above to get 244^{256}, then we reuse some of our previous answers and two more multiplications to get $244^{321} = 244^{256} \times 244^{64} \times 244^1$. This gives the answer with ten multiplications instead of 321. Even better – the bigger the power, the more significant the saving in effort and computing time.

Before looking ahead, you might like to try to program this method to calculate big powers. It's a good test of your understanding of binary numbers and powers in general.

```
def bigpower(base,power,modulus):
    ans = 1
    while power:
        if power%2:
            ans = (ans*base)% modulus
        power = power//2
        base = (base*base)%modulus
    return ans
```


```
def bigpower(base,power,modulus):
    ans = 1
    while power:
        if power%2:
            ans = (ans*base)% modulus
        power = power//2
        base = (base*base)%modulus
    return ans
```


The Python function above will implement this method and could be used instead of the built-in `pow` function. It builds the answer up while decomposing the power at the same time for added efficiency, so we don't need to store all of the intermediate powers. The `while` loop stops when `power=0`. The `if` statement only runs if `power%2=0`, this corresponds to picking out the ones in the binary version of power. (HW)

Mark Thornber has been a maths teacher at Durham Johnston for the last 25 years. Mark's first computer was the classic ZX81.

PHONICS KRISS-KROSS: PRACTISE SPELLING AND LOGICAL THINKING

Give KS1 pupils a solid foundation for writing programs as well as writing English – get them solving simple logic puzzles while practising phonics

STORY BY Paul Curzon

You do not best learn to write just by writing stories, and you do not best learn to program just by writing programs. There are many activities that help develop the building blocks for becoming a good programmer. Perhaps the most important to practise is logical thinking: an ability to reason about what is true or false, think clearly, and have good attention to detail. This is a widely useful skill that supports maths too. There are many logic puzzles that can make this fun, but one of the best kind to start with are Kriss-Kross. They also help develop an understanding of sequencing – doing things in the right order – vital to algorithmic thinking.

Kriss-Kross puzzles

Kross-Kross are simple word puzzles. You must fit given words into a grid. Solving them requires no special knowledge, just logic. As several words initially fit each slot, you must use clues from the words and the way they overlap. You cannot write the words in the grid in any order, but must follow the sequence dictated by the logic.

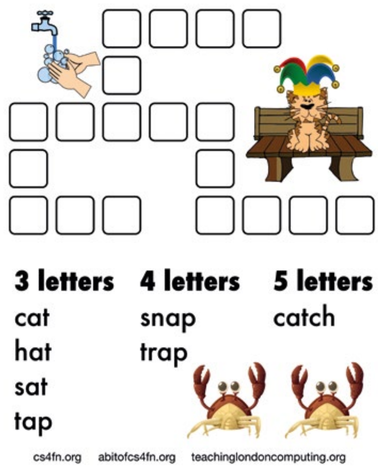
If there is only one word of a given length left, there is only one place it can go. Once you know some letters, then you can use those clues to find a word that fits those places. If only one word fits, then you can put it in place (and cross it off the list), but if two or more do, then you must look elsewhere first, until one of the possibilities has been ruled out.

Guided help may be needed to start, to ensure the pupils do follow the clues, checking that only one word fits a slot. Once they have the idea they can quickly move to doing them on their own, or better still in pairs, each with the same puzzle, saying the words, and explaining their reasoning to each other as they solve it.

Practise phonics too

Phonics Kriss-Kross are puzzles made of collections of similarly sounding words. I created the first as a fun way to help my five-year-old practise phonics beyond his writing-word-lists homework. He told me the words to use, and picked pictures of them. I put them in a puzzle, and he solved it. He liked doing it so much that (unlike the word lists) he demanded we do more, and then more. In the evenings, we chose words and pictures together. I put them in a grid overnight and left it by his bed. He found it in the morning and did it for fun before breakfast. As a result of his enthusiasm, there are lots of puzzles at **helloworld.cc/2qNSNK0** on easy-to-print sheets and as a booklet to give out.

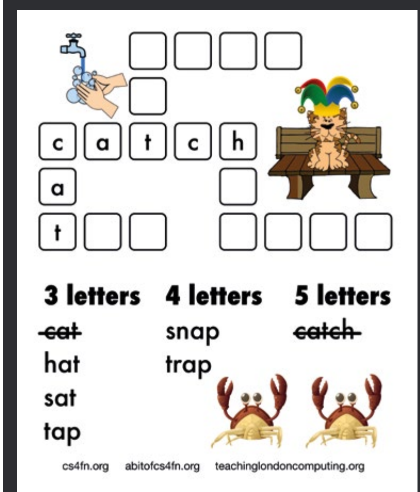
The more pupils practise skills like spelling and logical thinking, the better they get at them. The more fun it is, the more they practise. So why not make practising writing more fun than copying word lists by adding a dose of logical thinking? (HW)



An example Phonics Kriss-Kross. By Paul and Daniel Curzon, images from pixabay

SOLVING KRISS-KROSS

To solve the given puzzle, first notice only one word has length 5: catch. There is only one place it can go. Put it there. Now look to the clues it gives us. We need a 3-letter word starting with C. There are four 3-letter words, but only one starting with C: cat. That gives us a new clue ... and so on. Enjoy!



Paul has been a member of CAS since its foundation and is currently a CAS Board Member. He is a Professor of Computer Science at Queen Mary University of London and co-founded both cs4fn and Teaching London Computing.

CREATIVE INSPIRATION WITH IOT

How the Apps for Good IoT course inspired our students to get thinking creatively to solve important problems

STORY BY Paul Gallanagh

Our school canteen was strangely quiet during the period we worked on Apps for Good with our S1 pupils – with many of our young people choosing to spend their lunch times working together on their app ideas, supplementing the class time spent on this pioneering project. Not many school-based learning experiences can stake the claim of pupils sacrificing social times, but it was evident Apps for Good sparked something positive in our junior pupils from the get go.

Dunoon students with their winning prototype IoT idea which aids elderly people in the home

Although we have been teaching coding to the youngest of our pupils for over a decade, we had struggled to find a real and relevant context to capture imaginations and to ignite passions.

We were intrigued to hear of the new Apps for Good Internet of Things pilot learning programme, and were delighted that Apps for Good added us to be the list of schools taking part in this pilot. Our S3 (middle school) pupils have worked on the Apps for Good programme for the last few years, so we knew that our young people were in for a treat.

The Apps for Good recipe for effective learning was clear in the new IoT learning resources: team working, communications, problem solving, as well as coding were all key ingredients – but for us the creative forum offered throughout the learning experience was unrivalled.

Our pupils were commissioned with the task of conceiving, researching, developing, and producing a smart internet-enabled device that would have a positive impact on society. They certainly rose to the occasion and this, coupled with the extent they were willing to adopt and incorporate a social dimension to their work, was fantastic. From baby mobiles with in-built thermometers that would switch on a fan to cool down baby when the room got too hot, to sun hats with UV sensors that would notify a user, via a web app, to keep in the shade or to slap on more sun lotion. The range of ideas produced by our young people was innovative and inspiring, and testimony to the quality of the Apps for Good learning resources.

The wealth of resources and support from the amazing Apps for Good team at each stage of the programme was fantastic. Our teaching staff were no longer the sage on the stage, but more the guide by the side, as our young people embarked on a range of pupil-led initiatives. The great flexibility within the programme ensures it is accessible for all – for example, a range of coding platforms were used by our young people, to suit a range of abilities – including the BBC Micro:Bit block editor, Microsoft Touch Develop, and Visual Basic.

The opportunity for our young people to discuss their concepts with industry experts – easily organised via the Apps for Good portal – is a key strength of the process. Our young people were enthused with the encouragement and advice received from a range of experts, fuelling further work on their ideas.

Our time working on Apps for Good this academic year ended in spectacular fashion, with double delight at the Apps for Good UK awards in June. We were extremely grateful to receive the UK School of the Year award, and one of our teams scooped the inaugural IoT award for their SafeStep idea – a range of smart internet-enabled rugs that can sense if someone has fallen, and can then automatically contact a loved one for assistance. This recognition is a great honour for our school.

We are now looking forward to building on the successes of this year and are excited with the new developments Apps for Good are planning – including innovative assessment approaches. Apps for Good certainly do not stand still, and seek continuous improvement and evolution of their learning opportunities. We are thankful to have been part of their journey the last few years and we have warned our kitchen staff to expect further lean times shortly, when our new batch of S1 pupils have their first taste of Apps for Good. (HW)

CHRISTINE SWAN CAS MASTER TEACHER

GET INTO COMPUTER SCIENCE TEACHING

I didn't get the chance to study computer science at school and certainly never thought my hobby would become my main occupation.

I was a teacher for 28 years, 18 of those teaching computer science, and have been a CAS Master Teacher since 2012. I am now a visiting lecturer, author, presenter, trainer, external moderator, and proud member of the Raspberry Pi Foundation.

Do you love finding out how things work? Will not quit until you have solved a problem? Ever thought of teaching computer science? I suspect readers might answer yes to the first two questions, but might be stunned into silence by the third one.

I was always fascinated by technology and mechanical devices for as far back as I can remember. I used to help my dad fix the car by passing tools, while standing on the bumper. Dad also had a passion for electronics, fixing radios and televisions as a hobby, to the annoyance of my mum, and delight of friends and neighbours in need. I quickly learned all resistor colour codes and how to test components using a multimeter while I was still at primary school. Later, at university, I learned how to program in BASIC with a smattering of FORTRAN while studying for a degree in biology at the University of London. I knew I wanted to be a teacher, but if there were routes into computer science education in the early 1980s, I was not aware of them and, anyway, I wasn't a computer science graduate, so that was that.

Bring on the Beebs

I qualified as a science teacher and started teaching in a secondary school. Year 7 were lucky enough to do something different on Friday afternoons, when myself and another chemistry teacher taught BASIC on BBC Microcomputers. We did also have a RM380Z computer but as an NQT, I was told: "Don't touch!"

I explored what the BBC was capable of and pushed it to its limits. I wrote reports using them, managed our budget, wrote letters, ran simulators of nuclear power plants, but also controlled pneumatic systems and motors using interface boards. We had an Econet network to allow students to access programs stored on file servers, and heard about this thing called the internet via the National Educational Resources Information Service. Fast forward three decades, the Raspberry Pi has brought back creativity, innovation, and the desire to experiment with technology. Digital makers inspire to push the Pi to its limits just as we were doing from 1980 with the 'Beeb'. A key difference is access to shared ideas, support, and communities.

Are current computer science teaching skills any different to those pioneering days? I think not. Although I was fairly certain that I wanted to be a teacher, I've switched the subjects that I teach from the one that I graduated in. Technology evolves quite quickly, so it is very important that practitioners keep their knowledge and skills up-to-date.

Teaching in practice

There are a variety of different routes into teaching. As well as the conventional degree and PGCE combo, graduates can follow a School Centred Initial Teacher Training (SCITT) programme or, if you are already working in a school, can achieve QTS via the assessment only route. However, if you are an enthusiastic geography teacher who'd like to teach computer science, there is lots of support on the Computing At School (CAS) website, and you can become a Raspberry Pi Certified Educator via the Picademy programme. Colleges and universities run Subject Knowledge Enhancement (SKE) and CPD courses which allow non-computing graduates (like me), to cover all key topics on GCSE and A level specifications. How do you know if you want to teach computer science? Get involved with extra-curricular clubs such as CoderDojo or Code Club as a volunteer. The most important requirement is enthusiasm and a willingness to never stop learning. (HW)

Christine Swan is a CAS Master Teacher, visiting lecturer in Computer Science education at Birmingham City University, author, and presenter.

GETTING STARTED AS A COMPUTING EDUCATOR

James Robinson describes the steps that he went through in transitioning to teaching computing, and the key lessons he took from the process

STORY BY James Robinson

Back in 2011, I began to make the transition from teaching other subjects to becoming a computing educator. Computing was gaining traction as a subject in UK schools, a shift in focus that I found hugely exciting, and I could see the opportunities and appeal that computing brought with it. As a child, I had a quarter share in an Atari 65XE on which I began writing BASIC programs – and here my interest for computing began. I continued to tinker throughout my school years, but as computing wasn't an option at school, I had no significant programming experience when I started my degree in Computer Science. I emerged from university with a firm theoretical knowledge... but somehow still with fairly minimal confidence in programming. It was with this background that I began investigating how I might bring computing to my secondary classroom.

Extra-curricular learning

The first step was something that I realised I'd already been doing for ages, and that was running extra-curricular sessions for students. Since I started teaching, I'd been running clubs at lunch and after school, including Lego Club, Computing Club, and a few seasons of the robotics competition First Lego League. Working with my students in an informal context was great for them, but also great professional development for me. I tested out projects, trialling new ideas and activities outside of the pressure of a formal lesson. These sessions were opportunities for us to make, fail, play with ideas, and learn together. I learnt a lot about what works, improved my own knowledge and understanding of the programming skills needed, and gained a real understanding of what the students enjoy, what engages and motivates them. In addition, the students attending often became some of my greatest assets as I introduced activities to more formal groups. Many educators empower similar groups of students, giving them roles such as 'digital leader', with the ability to help set up kit or support other learners.

KEY LESSON: Using informal, extra-curricular settings to gain experience, you'll be better equipped to explore computing concepts in ways that connect with your learners. There are so many great opportunities out there to start an extra-curricular club, all with fantastic support. Many of the educators that the Raspberry Pi team train go out to start clubs as a first step in their computing journey.

Physical computing is a great concrete way to introduce learners to new concepts and skills

Connecting with other educators

Early in my career, I'd been along to a few teacher meet-ups, run by an organisation called Computing At School, who had the aim of reintroducing computing back into the UK education system. I don't remember much of the content from these early sessions, but I do remember how great it was to connect with other educators and hear about issues they faced and share solutions with one another. So important, in fact, that I went on to support my local meet-up and eventually take on the responsibility of running it for a few years. Since then, I've sought opportunities to connect whenever possible through events, conferences, training programmes, and online. Establishing a network was important to me because it allowed me to share, learn from others, and kept me motivated. Even now, whenever I connect with one of these networks to get advice, share an idea or seek feedback, I always come away enriched.

KEY LESSON: Teaching can be an isolating experience at times - particularly subjects such as computing that often only have one teacher in a school. Finding others to support you is crucial - there are many networks you can join. Take every opportunity to connect and share.

Learn something new

As a teacher, I've attended a wide range of training events, and seen many examples of both good and bad training. There are now loads of courses out there and ways to learn; however, finding something that's worth giving up your time for can be tricky. Early on, I had decided that I wanted to use Python as the main language I was going to teach with, it was easily readable but also hugely versatile – the one problem was that I'd never written a line of Python! Once again, I turned to my trusty network for recommendations, and found a great course run by Dr Sue Sentence and Adam McNicol called 'Python School' (**pythonschool.net**), which was a series of two-hour sessions held at a local school. The sessions were great and gave me a solid grounding in using Python, and there was homework to ensure we continued on our own, and content that was adaptable and extendable. Sometimes, half the battle of learning something new is finding the time – the great thing about regular courses is being able to schedule the time to learn each week. Another training highlight of mine was attending a Picademy run by the Raspberry Pi Foundation, which had me returning to school so inspired and with a head full of ideas. What both these courses had in common was their hands-on nature, learning through doing, and making mistakes along the way. Picademy, especially, made me introduce a more student-led and engaging style of teaching with my classes.

KEY LESSON: As educators we have precious little time and headspace to learn new skills, so it's important that we seek out the best training available whether it's online or face to face. Get recommendations from your network and make time to learn.

Lead by example

As I've said, I had my first programming experience at about age seven, and from that moment I've been passionate about technology and making things with it. Even before 2011, I would always share news stories, gadgets, and interesting projects with my classes, squeezing computing concepts in around my existing curriculum. I made things that I wanted to make and demonstrated them to my students – tried things, failed lots, and shared it all with them. I hope that some of my passion and enthusiasm rubbed off on at least some of my students.

On occasion I've heard other educators say, in front of their students, things like "I'm no good at programming," or "This isn't for me". This upsets me as you wouldn't hear the same thing said of literacy, history, or problem solving – especially when (as my five-year-old daughter will insist) the simple addition of the word 'yet' on the end can change this negative message into something wholly more positive.

KEY LESSON: Those of us privileged enough to help shape our students' understanding need to positively embrace these challenging new skills. We ourselves need to become makers, programmers, and role models for our learners. We have to learn alongside them, be motivated by things that interest us, and model resilience to failure.

Baby steps

All of this might sound quite daunting, what with all this leading and being inspiring, but it needn't be. All it takes are small, iterative steps. The first computing lesson I taught after Picademy was literally an hour of talking about the one device I had, passing it round and talking about how we might use it. The following lesson we connected all our PCs to it and interacted with it using written commands. Next, we wrote our first Python code as a group, over a network, on a Raspberry Pi. We shared this success with other teachers (including leadership), and managed to get some funding for more kit, which allowed us to do more, demonstrate more, and get more funding and support. Each new lesson I taught, I learnt something; about computer science, about my classes, about how my pedagogy needed to develop – and every following lesson I tried to apply what I'd learnt. Most of the time, I was barely ahead of the students and remember a particular project I'd planned, in which we were going to make the game 'Guess Who' with the Raspberry Pi camera. A couple of weeks later, I realised we'd bitten off more than we could chew and cut the project short, but had learnt loads along the way about file access, lists, and GUIs.

KEY LESSON: We can't do everything at once. Computer science is a vast and new subject for many of us. Start small, reflect, iterate, and repeat.

Share everything

Finally, one of the last things I learnt in becoming a computing teacher was how important it is to share. I started talking about some of my ideas and resources at meet-ups and events. I found that I not only got a sense of pride in something that I'd made, but others could benefit from it too. Best of all, other people suggested how I could improve them. Just before I left the classroom I became a 'CAS Master Teacher', which meant I planned and led training for other teachers in the area. Every session I ran, I would help other educators but also learn something myself. In my work for the Raspberry Pi Foundation I get to meet a lot of educators, some of whom have shared the view that they're 'just a teacher', or that they have nothing worth sharing. This of course is nonsense – every educator out there has lots of valuable knowledge, experience, advice, and resources to share. Whether you find opportunities to train your peers, speak at events, get involved in discussions, open your resources to others, or even write for this very magazine, you should be sharing. (HW)

KEY LESSON: Whatever point you're at on your computing journey, there's always more to discover; all you need to do is keep making, learning, and sharing.

FEATURE

Our students, delegates and teachers

Images courtesy of Flo Capper Photography

CELEBRATING WOMEN IN COMPUTER SCIENCE AND ENGINEERING

International Women in Engineering Day 2017 was celebrated in style by King Edward VII School in Sheffield

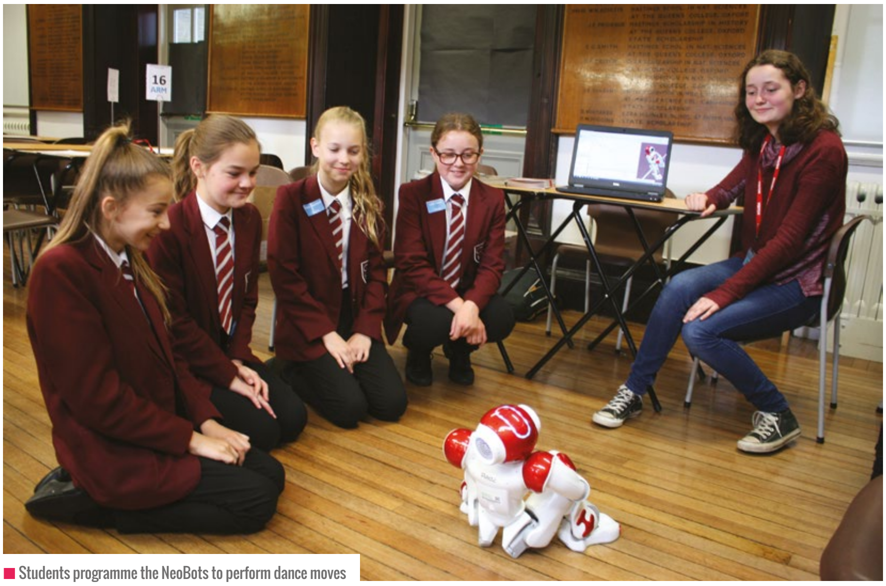
STORY BY David Kavanagh

Disadvantaged female students from South Yorkshire attended an exciting day at King Edward VII School in Sheffield to help celebrate International Women in Engineering Day 2017. The 80 students were joined by female computer scientists and engineers from all over the country, from industry and academia.

Organising the event

Our event was fully sponsored by ARM UK. Gaia Innovation were tasked with organising the event, contacting the schools involved, and all the visiting speakers and academics. I met Leisha Marlow from Gaia Innovation at an Open Day at Factory 2050 in Sheffield earlier in the year. She explained what her company did, and I asked her to help me organise my event, aimed at addressing the lack of opportunities for girls in computer science, with a particular emphasis on disadvantaged students. I already had ties with ARM, as I had taken students to their offices in Sheffield for their Ada Lovelace Day celebrations, and subsequently organised Year 10 student work placements there.

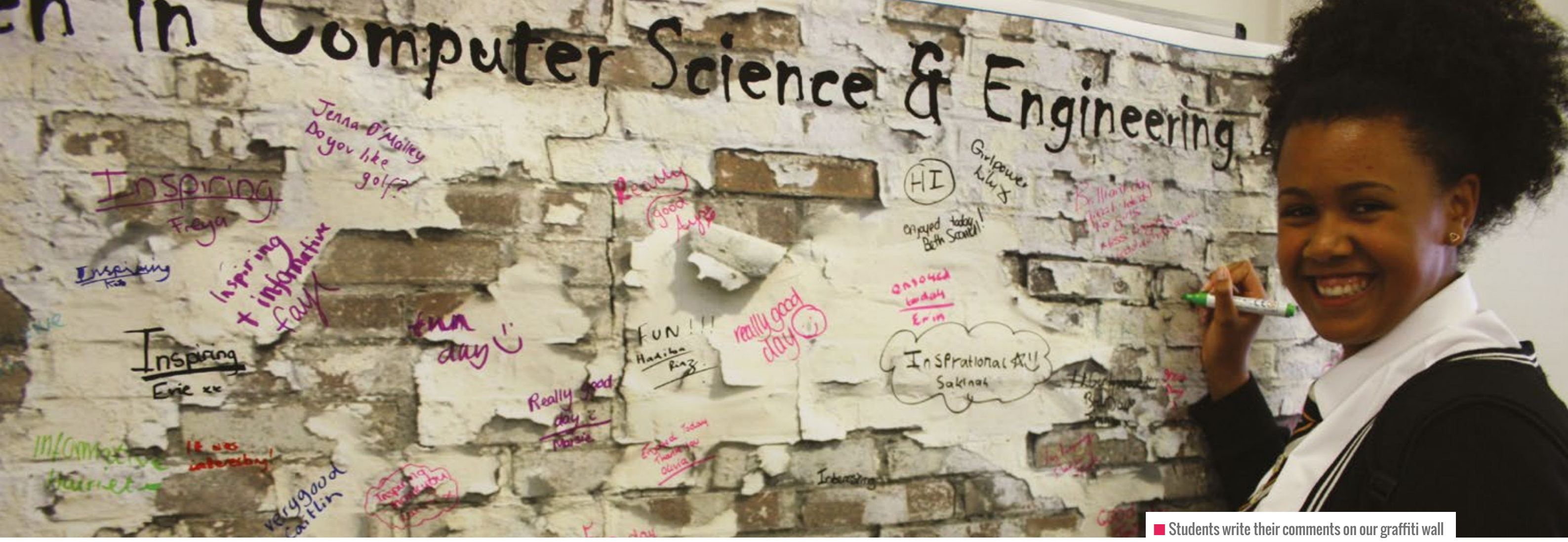
Six schools took part in our event: Trinity Academy from Doncaster; Rawmarsh Community School and Wickersley School and Sports College from Rotherham; and Meadowhead School, King Ecgbert School, and King Edward VII School from Sheffield.



Students programme the NeoBots to perform dance moves

A warm welcome

The day started with a warm welcome to all the students, accompanying staff and delegates. This was followed by some inspirational speakers, including Maha Khan, an engineering graduate and current Graduate Consultant at ARUP; and Andrea Catlow, Site Operations Director at Frontier Pitts. The students were invited to take part in four computer science and engineering-related activities: 'Mission to Mars' using



Students write their comments on our graffiti wall

ChickBots and various sensors to locate water and heat and to avoid obstacles; 'Rocket Launcher' using BBC micro:bits and K'NEX to build and set off a rocket; 'Solar Race', where the students built and raced solar powered cars; and The University of Sheffield brought their ever-engaging NeoBots for the girls to program. The final part of the day was a Careers Speed Networking event, where groups of students spent ten minutes with each inspirational speaker, listening to their stories, and asking questions on the careers and academic courses available to them.

An informal lunch in the school library followed, for all the delegates and visitors, where they had an opportunity to relax and chat after a thoroughly busy and exciting day.

> "THE STUDENTS WERE INVITED TO TAKE PART IN FOUR COMPUTER SCIENCE AND ENGINEERING-RELATED ACTIVITIES

Inspiration

The idea for the event came from conversations I have had over many years with female computer science students. Many of them were fantastically able and adept, but they had not gone on to choose computing at GCSE or A level. This led to very small numbers of female students considering degrees in computing-related subjects. It became clear that they were being discouraged by a lack of opportunity to explore the subject without the boys 'taking over', and a lack of knowledge of the courses and careers available to them as computer scientists. There is much research that backs up this hypothesis.


Programming ChickBots to find water


Solar-powered cars - students could change the gearing and wheel size to make them faster

Legacy

The day itself was a big success. Initial feedback from students, via our graffiti wall and subsequent Survey Monkey research, was overwhelmingly positive. The majority of girls commented that they had 'had their eyes opened to the many opportunities available to them' and that they had had 'no idea how much fun it was'. Feedback from the delegates was equally enthusiastic. One delegate commented that 'the level of engagement was amazing'. Andrea Catlow, who had driven all the way from Brighton that morning (and was driving back that afternoon), commented that during the Careers Speed Networking event, she was 'thrilled to see so many students genuinely interested in her story and how her career had developed, and how they might follow in her footsteps'.

Many thanks to all those involved who helped to run such an amazingly successful day. I am particularly delighted, as I was hoping to run it again next year – and on this evidence, I will definitely go ahead. (HW)

David Kavanagh
Curriculum Leader for Computing and IT, King Edward VII School, Sheffield ZX81.

PROGRAMMING: TAPPING INTO I.T. PROFESSIONALS

Is there a place in the classroom for programming professionals and, if so, how we can engage them?

STORY BY Richard Gray

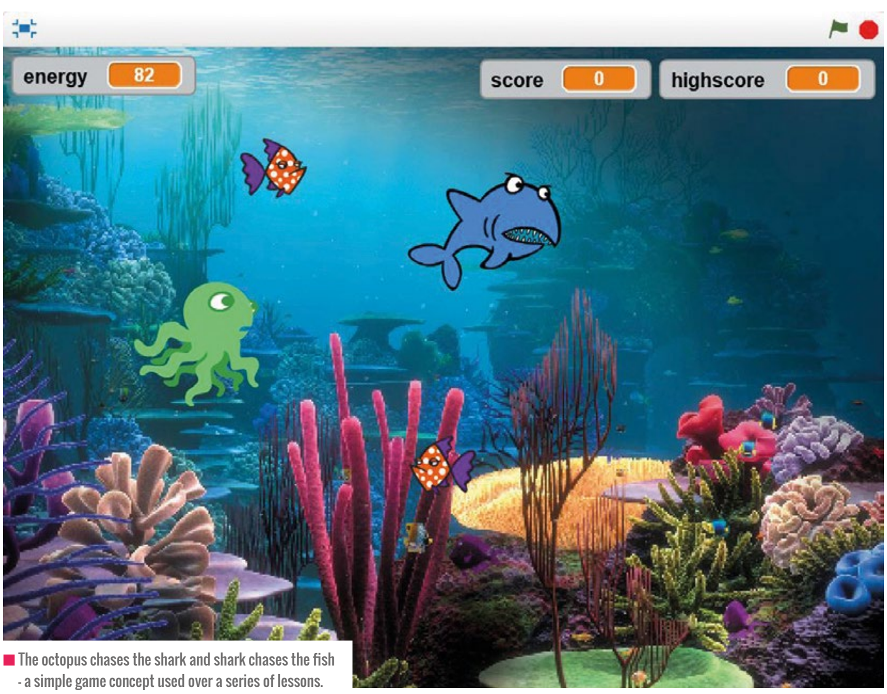
A hand shot up from the other side of the classroom, "Mr Gray, I need some help!"

For the past 18 months, I've been teaching Years 5 and 6 how to code. Whether in small groups or whole classes, I was in my element – but I'm an IT professional, not a teacher.

I'd spent many years training for my profession and learning how to program. For many primary school teachers with already heavy workloads, it had always felt more than a little unfair to me, having to ask them to learn about something so very new, and in sufficient depth to feel confident teaching effectively.

I felt very strongly that I wanted to do something to help those who felt somewhat freaked out at the thought of programming, but I also wanted to ensure that what we were teaching would have a grounding in the real world and, at the same time, give the pupils an insight into what we, as IT professionals, actually do all day.

I'm fortunate enough to work part-time, and I had some 'spare' hours to put to good use. I spoke to my local primary school and they were supportive.



The octopus chases the shark and shark chases the fish - a simple game concept used over a series of lessons.

Real-world lesson planning

At the start of the last academic year and, in conjunction with the Year 6 teachers, we planned four terms' worth of lessons – one hour a week per class. We would work on building an educational computer game for Year 3 students, but would go through the full project life cycle – discovery, market research, planning, design, build, testing, evaluation, and presentation.

Pupils' exposure to Scratch in earlier years was limited, so we spent the first weeks familiarising ourselves with the interface, creating simple programs to demonstrate basic concepts such as loops, sprite movement and mouse/keyboard input. We then moved on to look at variables and collision detection.

As a whole class project (working in small groups of two and three), we developed a simple game concept over the course of a term – the player moves a shark character around the screen eating computer-controlled fish (point in a random direction and move, bouncing off the screen edges). They avoid being touched by the deadly octopus who follows the character and drains their energy. Add a high-score table and you're done.

I was able to teach best programming practices, discuss the importance of good variable names, show how to break a larger program down into smaller elements, run practical debugging sessions, and reinforce pupils' understanding of event broadcasting, Boolean logic, looping, and control structures.

I've written many games in Scratch, and I'm comfortable fielding questions about what is, and what is not (at their level) possible, so when we started our main project (making their own games), we encouraged them to explore their own interests and ideas as to what made a 'good' game and then developed these into something playable.

The research, development, build, test, and evaluation processes we went through mirrored those in the real world and it felt 'right' to be taking the children through these same steps. As a published educational game developer (The Shattered Crystal), I was able to talk to them about the different roles involved in creating and producing a game, and I felt it important to emphasise that it wasn't just programmers who were critical to success – level designers, graphic designers, musicians, project managers, advertisers, marketing specialists, and myriad other roles all combine to create the finished product.

For the teachers, they have a second adult in the classroom to help deliver the lesson and work 1:1 with small groups. There are opportunities for learning alongside the children and to add to their CPD portfolio. I would hope that they too would feel comfortable enough to ask questions of their own to further extend their own programming knowledge. Lessons can be tailored to tie in with class topics and, at the end of the course, there is a whole set of re-usable lesson plans and resources.

Looking to the future

Whether there will be enough people with the interest, time, and motivation to volunteer to make this a reality remains to be seen. Would ever-tightening school budgets stretch to accommodate buying in of this resource, in the same way that PE coaches, French or music teachers can bolster teaching in their respective subject areas? If, as we're told, we're heading towards a coding skills shortage, could the government provide funds for small businesses to release programmers for a couple of hours a week to support teaching staff?

Whatever the future, I would strongly recommend teachers reach out to parents and local businesses to see whether opportunities to engage IT professionals might exist. Until I was given the chance to stand up and teach, I didn't realise how fulfilling and rewarding it would be. And the involvement of your IT professional can vary from just being in the classroom answering questions to, like me, delivering whole lessons with the support of the class teacher.

I have loved seeing the children learning about programming, being motivated by working on projects that interest them personally, overcoming challenges, persevering, asking questions, achieving a working game, and running through the project life cycle of design, development, testing and deployment – all with a real-world focus.

Everyone achieved something that year. Everyone made a game. Everyone understood what they were doing and everyone exceeded our expectations – mine, their teacher's and, quite possibly, their own. (HW)

> " I WANTED TO DO SOMETHING TO HELP THOSE WHO FELT SOMEWHAT FREAKED OUT AT THE THOUGHT OF PROGRAMMING

I hoped that, in some small way, my work with these classes allowed them a glimpse into the future – into the real world – and to see that what they had achieved in class was something to be proud of.

So who benefits?

In an ideal world, I would love to see IT professionals working alongside class teachers, bolstering confidence (in both staff and students) and encouraging open-ended creative investigation and questioning through making something fun – computer games.

The pupils see a fresh face in the classroom, have opportunities to ask in-depth programming questions, and can work on open-ended projects which personally interest them and are tailored to their individual needs and preferences. They receive the benefit of professional programming guidance and real-world input.

A TEACHER'S PERSPECTIVE

"As a primary school teacher, we have a relatively shallow knowledge of the foundation subjects and so having someone who knew the subject well, knew what the misconceptions could be, who knew where to take the programme and how to extend the more able children was the prime benefit for me.

As well as the time in the classroom, to have someone who could supply resources (sample programs) that were pitched at the 'right' level for my class, and respond to changing needs and levels of understanding was something I welcomed, as was having assistance with planning and preparation.

With the best will in the world, we don't have time to investigate all the foundation subjects in the depth we might like, and even using resources from the internet (many of which are of questionable quality) I would still have needed to spend a great deal of preparation and planning time which, arguably, would be 'better' spent on subjects such as maths or English.

Looking back, I was amazed at how well the class responded to having someone else take their lessons - they loved that fact that Mr Gray wasn't a teacher, but a 'proper' computer programmer. They were incredibly engaged throughout the year and really enjoyed each session.

When the class started to make their own games, having someone who could work closely with the groups was superb - everyone received a personalised level of support beyond that which I would have been able to give. I just wouldn't have had the time or knowledge.

If I had my way, I would have this help every year and would recommend it to colleagues in a heartbeat!"

Year 6 class teacher

BLUFFER'S GUIDE TO PAIR PROGRAMMING PEDAGOGY

There is far more to Pair Programming than simply asking your students to work in pairs. In this helpful guide, Alan O'Donohoe, computing teacher, leader of exa.foundation and an advocate of Pair Programming, wants to convince you to try it in your own teaching, so he has provided you with a range of strategies and advice you can use to great effect in your own classroom to unleash the great learning potential within this approach

Why should I consider using Pair Programming?

From 2010 to 2015, I embarked upon my conversion from ICT teacher to Teacher of Computing. During this transition period, I experimented with a range of teaching and learning strategies, including techniques like rubber duck debugging, Sabotage, and pair programming. I developed a handful of teaching strategies that I believe significantly improved the manner in which I had become accustomed to teaching. These strategies didn't just guarantee more successful outcomes for my students, I felt they also liberated me from the most onerous and stressful activities, like debugging students' code.

This meant I could more objectively evaluate the learning taking place and consider how I might plan the most effective teacher interventions. Of all the strategies that I tried, I found that pair programming had the largest impact on supporting pupils learning how to programme, design algorithms, and think like a computer. Many other teachers have also successfully converted to pair programming pedagogy.

Richard, teacher: "I tried it yesterday… Wow! What a difference it made to a difficult class. With a couple of notable exceptions, the class were engaged, discussing, and most of all programming, differentiated problems according to the ability of the higher level student."

What about traditional teaching methods?

When I first started teaching in the early 1990s, I adopted a teaching paradigm that firmly placed me as the 'expert in the classroom'. When I started teaching Computing I soon realised that this was unsustainable. My GCSE classes were asking me questions that I had not yet learned the answers to myself, and they increasingly relied on me to debug their coded solutions for them. At first, it inflated my ego and self-confidence to know that my classes were heavily dependent upon my knowledge and experience, this resulted in a constantly high demand on me to support them – but soon I found this tiring and stressful. There were some students I could never remain a few steps ahead of.

Through pair programming, my students developed greater independence – relying less on support from their teacher, they collaborated more among themselves using each other as a resource, and their levels of engagement were far higher than

Pair programming has a large impact on supporting pupils learning how to programme

PAIR PROGRAMMING CHALLENGES:

1. Create a program that will ask the user to enter two separate numbers, then print the output like this, 'You entered two numbers n1 and n2.', where n1 is the first number, n2 is the second number.
2. Create a program that will ask the user to enter two numbers, then add them together and print the output like this, 'You entered two numbers n1 and n2. The sum of these two is n3', where n3 is the sum of both.
3. Create a program that will ask the user for two numbers, then multiply them together and print the output like this, 'The product of n1 and n2 is n3'.
4. Create a program that will ask the user for two numbers, then divide the largest number by the smallest and print the output like this, 'When n1 is divided by n2, the answer is n3' where n1 is the largest number, n2 is the smaller number and n3 is the answer.
5. Create a program that will ask the user to enter two numbers, then ask the user if they are to be added, multiplied, subtracted, or divided. It will then perform the calculation for the user as above. Add some validation to your coded solution so that your program will only accept integers from the user input.

any other previous strategy I'd tried. I discovered there were also significantly fewer bugs to solve, partly because there were always two pairs of eyes on the coded solution, so they stopped asking me to debug their solutions.

It's worth stating that I didn't experience the positive benefits straight away – far from it. I needed to persist, observe, and reflect on what was happening, I had to relinquish some traditional control in order to allow my students to develop in confidence, and let go of some long held beliefs and habits of mine which had become embedded.

Won't the teacher become redundant?

Not at all, the role of the teacher develops into a far more useful one. I know that developing paired programming within my own pedagogy had a positive transformation on pupil learning outcomes. Previously, I fell into the trap of thinking that I needed to have mastered all of the Computing curriculum and the entire programming language before I could teach it effectively. I used to believe that the only path to my students' enlightenment was through me. However, in order for our students to grow and develop, then part of that process must require the teacher to create conditions in which students can flourish without direct intervention from the teacher.

What is the best way to try pair programming?

I'd recommend you start as soon as possible, which will allow you to develop the strategy in your own class. In terms of ability, pair like with like so that the stronger, more confident students are paired together and the weaker students are also paired. Explain that each pair will only develop the code on one computer throughout. One partner drives (with the keyboard and mouse) and the other navigates (vocal instruction and feedback). Then every five minutes or so, tell them to swap roles. Once you've started this with your classes, you'll spot ways that you can improve and develop the use of the strategy. You'll be surprised how few demands a class make on you once pair programming has become established with a teaching group.

In the rest of this article I describe some of the 'pair programming' strategies that I used, which I sorely wished I'd discovered earlier on, plus we'll include some further reading. If you wish to harness the full potential of pair programming in your classroom, here are some guidelines to follow:

Clearly define and enforce roles (Driver & Navigator)

For paired programming to work effectively, it's important to clearly define the expectations of each role. For a fixed interval of time, the 'Driver' uses the keyboard and mouse, while being guided by their 'Navigator'. The navigator acts as both coach and mentor, guiding

Pair like with like for the best results

Ask students to swap roles during challenges

the driver along the right path. While the responsibilities of driving can be more stressful and intensive than navigation, partners swap roles every five minutes to share the burden and opportunities. Partners either swap seats every five minutes, or simply pass the controls over to the navigator.

Avoid misconceptions about pair programming

Some teachers remain to be convinced about the learning potential of pair programming. At first, it seems ludicrous requiring students to share computers when there are enough for every student to work on their own in isolation. Some teachers worry that their students might only produce half as much code. Well, if you're really lucky, your students might generate even less than half as much code! If you're not sure why this would be a good thing, read back over this paragraph again.

Share challenges and sub-challenges

Each lesson would start with one large, shared challenge for the group that I designed to address particular learning objectives, but decomposed into smaller sub-challenges. For example, if I identified that students needed to develop their experiences of file handling, I would design a task that required a solution that relied on file handling to some extent in the success criteria. Often the challenge would be an extension or development of a previous programming challenge.

Plan challenges with graduated difficulty

Since the students are likely to have different backgrounds, experiences, and levels of competency in programming, I chose to design challenges and sub-challenges that initially required all pupils to develop a solution to a relatively simple problem. The first sub-challenge may be an extension of a recently introduced concept. These sub-challenges would then increase in degree of challenge. Most pupils would be expected to develop a more challenging solution, and I would also describe a much more challenging extension that I expected only a small minority of the group could achieve within the available time.

Plan your seating carefully

The seating plans I used initially, paired pupils with each other according to their position in the register alphabetically. However, once I had some acquired some assessment and progress data, for example from the MCQs we were using, I was then able to pair students with peers where there was a small achievement differential. The first time

I tried this, I paired students from extreme ends of the achievement spectrum – but I found this to be counter-productive and led to other problems. It was only after trying this that I decided pairs matched by ability was a much more successful strategy.

Provide documentation and supporting resources

If there are particular resources that your class will find useful, share links and references to these so that the navigator can refer to them when appropriate. You might develop a resource bank to support all challenges that would include supporting documentation, links to similar solutions, and code snippets. If there were resourcing gaps, I often asked my students to search for resources and then add them to a shared document for everyone's benefit.

Teacher Interventions

During the pair programming sessions, I would typically visit each pair, speak to both partners, and initially praise them on something e.g. their use of comments, their efficient use of code. Then I would also suggest one thing I would like them to improve. This process allowed me to track the progress of different pairs, to spot any problems ahead, and to plan any whole class interventions.

Timed intervals

Typically students are instructed to work in each role for five minutes and then swap roles. I then planned whole class interventions every 15 minutes or so as required, to remind them about certain points, rules, to remind them about the characteristics of great navigators, or to share some solutions and approaches I had observed. The stopwatch feature on my phone allowed me to set an alarm at the end of each block. You could tell students to create a Scratch project that alternates between 'Driver (right) Navigator (left)' and 'Driver (left) Navigator (right)' every five minutes and plays a sound to remind partners to swap roles.

FURTHER READING:

- James Franklin's page on Pair Programming: **helloworld.cc/2D16pHH**
- CSTeaching Tips: **helloworld.cc/2DkuMgq**
- Laurie Williams' page on Pair Programming: **helloworld.cc/2D2a3jG**
- Pair Programming in a box: **helloworld.cc/2Fso8WO**
- Pair Programming Wiki: **helloworld.cc/2mjRyg9**
- Pair Programming video: **helloworld.cc/2D4qpZi**
- exa.foundation resources: **helloworld.cc/2DOHh3I**

Pairing students at opposite skill levels was not successful

> Some teachers worry that their students might only produce half as much code. If you're really lucky your students might generate even less than half as much!

Show & tell

There is terrific learning potential in sharing coded solutions with groups. After 20 or 30 minutes, I instruct the current driver to stand up and sit down with another navigator. Then each visiting driver has three minutes to review the code in front of them, suggest what works well to that navigator and what they might consider to improve their work. Then, when the visitors return to their original partners, they have lots to tell each other about what they have seen and heard from the others. The next time you repeat this, make sure to send the other partner to visit to keep it equitable.

Espionage

When there was an uneven number of students in the class, I introduced espionage. I identified a sensible student and asked them to choose any pair they would prefer to work with. Then, in addition to driver and navigator, I gave this group another role – 'spy'. When it was their turn to be spy, they had to wander the room stealthily looking for ideas to steal from other pairs, and then report back to their partners what they found while spying on other pairs.

Encourage and celebrate effective navigator behaviours

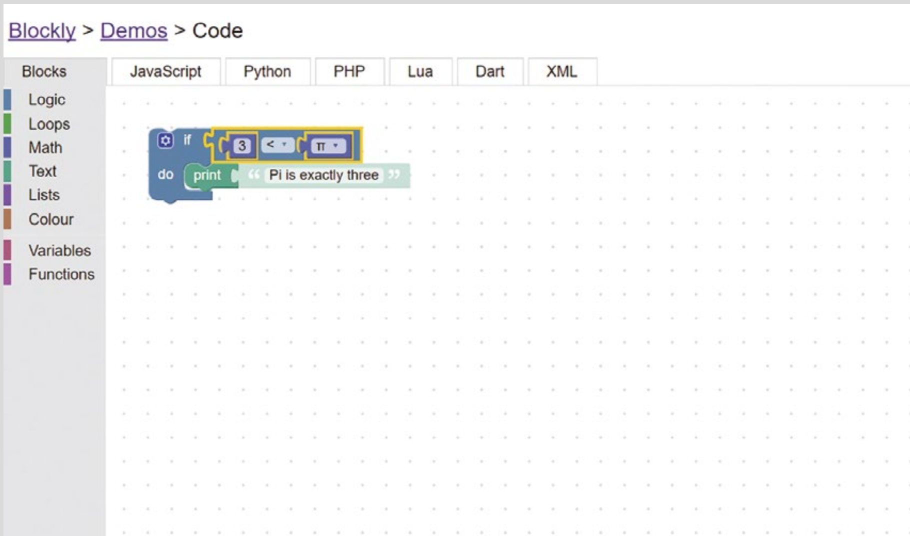
It's unlikely that you'll need to remind drivers what they are supposed to be doing, however being an effective navigator requires great skill. During my intervention I focus on positive behaviours I see navigators display. We had a poster, that the class helped me create, that lists the key behaviours of an effective navigator that included: be positive and encouraging; use clear, helpful language; look ahead to spot hazards and obstacles; look for ideas elsewhere; make suggestions... (HW)

VISUAL PROGRAMMING APPS

Teach coding with these visual programming languages

We don't need to tell you how amazing Scratch is for teaching. Its success has inspired the development of many more visual and block-based coding apps for a range of skill levels. The range is so broad that you could work your way up from Scratch to making megabucks on indie video games, using only visual programming languages.

Here are some of the best of these visual languages and supplements for use in education. (HW)



BLOCKLY

INFO **FROM** Google | **PRICE** Free | **URL** helloworld.cc/2tWI2lK

This app is a bit more advanced than the others, as it requires you to actually build your own blocks. The plus side is that it translates easily to a range of programming languages, such as PHP, Java, and even Python. It can be used to create your own version of EduBlocks (see page 88) that teaches, or allows you to translate into, other languages.

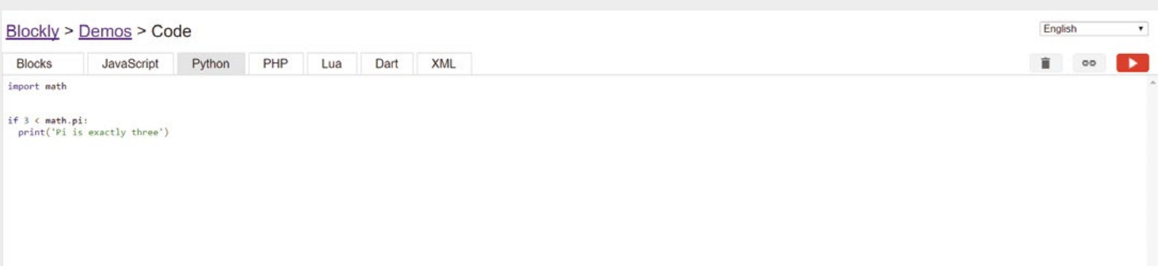
There are two use cases here. The first would be to create a more focussed visual coding app for a PC or smart device that students can use under certain conditions. You could even create a multiple-choice-style coding quiz. The other use case is to ease students into more traditional coding via building their own blocks and seeing how they interact; for example, building the necessary blocks to recreate previous Scratch programs.

At the end of a project using Blockly, you can easily output the blocks as code in your chosen language. In fact, it is able to translate code on the fly, so you can watch how the blocks change into the code in real time. This is an invaluable tool for translating block code into written code, while still being a very powerful block-coding tool in its own right.

Using Blockly does require you to spend more time on preparation, whereas something like Edublocks is ready to go after a quick setup. If you have the time, the payoff can be amazing. (HW)

QUICK FACTS

- Create your own block-based programming language
- Made by Google
- Can output real code on the fly

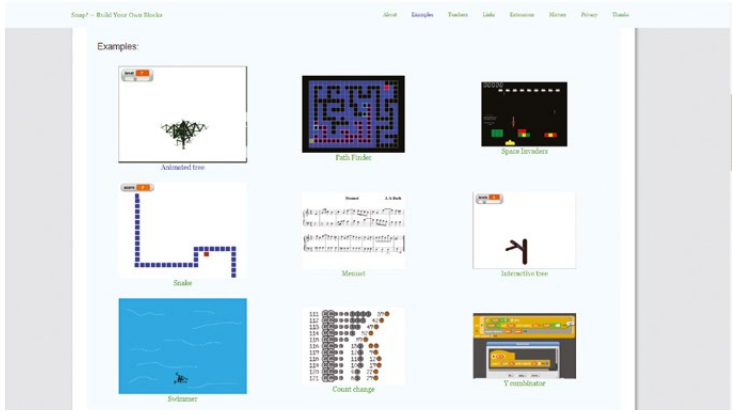


This image and above: the same code viewed in Blockly and in Python

QUICK FACTS

- Originally a Scratch mod
- Create your own blocks
- More advanced than Scratch

SNAP!



INFO

FROM UC Berkeley
PRICE Free
URL snap.berkeley.edu

Outgrowing Scratch is common among creative and imaginative kids, but not all of them are ready for a full-on programming language. Snap! is a great way to give learners more power, without taking away the familiar interface they are used to.

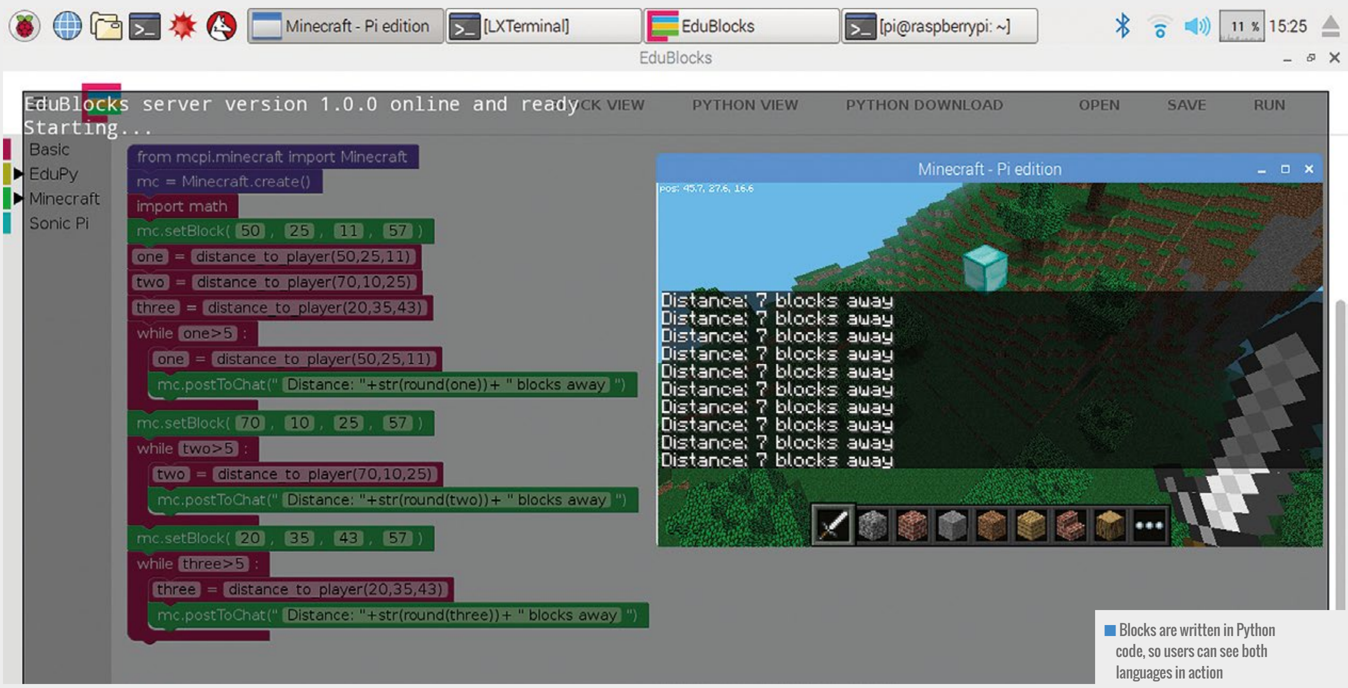
One of the things that Snap! offers is better Loop and If statement functions, as well as the ability to create your own blocks. It was developed at UC Berkeley, and is available free as an online IDE.

You can share and see examples in a very similar way to Scratch, making it an easy step up for the students who need something more challenging.

Creating custom blocks is easy, allowing students to mix and match existing parts of different blocks into their own unique blocks. They won't need to use a text-based programming language to do this, so no additional experience is required. This is great for anyone questioning why they have to repeat complicated blocks for seemingly simple tasks, and can even allow students to create brand new functions they may not have been able to use in Scratch.

Using Snap! is just as easy as using Scratch – you can tell that it started as a mod for Scratch, originally aimed at increasing the number of functions available. It does more than that behind the scenes, but for the user it feels like a more advanced version of Scratch. (HW)

Snap! feels like an advanced version of Scratch, with lots more programming power



Blocks are written in Python code, so users can see both languages in action

EDUBLOCKS

INFO **FROM** All About Code | **PRICE** Free | **URL** helloworld.cc/2tW5C2d

The main purpose of EduBlocks is to act as a bridge between Scratch-style block coding and more traditional programming languages, in this case Python. Blocks are written in Python code so you can see their real Python functions. Python is a good basis for a visual language, as its function names are fairly straightforward, and well-written Python code can usually be broken down into blocks anyway. This makes it easy to read and work out how it functions, which is particularly useful when moving on from something like Scratch.

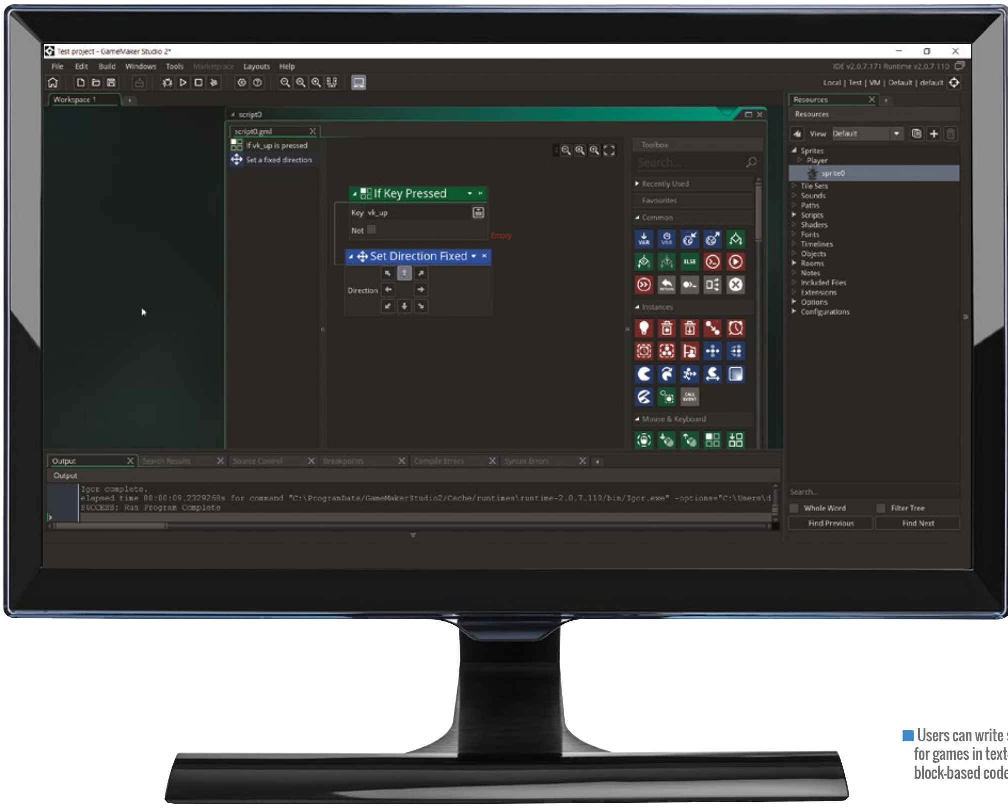
Examples and documentation aren't as prevalent as they are with Scratch, however it should be easy to adapt existing simple Python tutorials to work in EduBlocks. There is a free Minecraft resource available that provides a basic understanding of how EduBlocks works, and should give you enough knowledge to be able to craft your own lesson plans.

While EduBlocks does work in a browser, you'll need to have it installed on a Raspberry Pi on the same network to access Minecraft Pi. Depending on your network, this may limit its uses in a school environment. Code can be stored in the cloud, so students can work on projects from home if they wish.

At the end of a project, the blocks can be converted into pure code, demonstrating how the blocks relate to the syntax of a text-based programming language. (HW)

QUICK FACTS

- Transition from Scratch to Python
- Install on a Raspberry Pi, access from anywhere
- Control Minecraft Pi



Users can write scripts for games in text-based or block-based code

QUICK FACTS

- Used to create successful indie games
- Free trial version available
- Works on multiple devices

Tutorials and demos are available to help users get started with GameMaker

GAMEMAKER STUDIO

INFO **FROM** YoYo Games | **PRICE** Free trial | **URL** helloworld.cc/2u1ezez

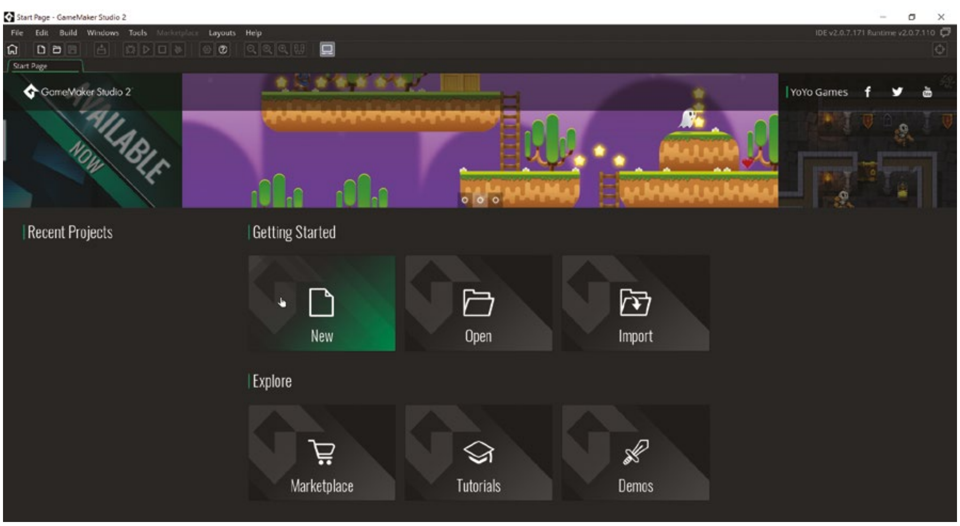
This is a much more advanced piece of technology, which has been used to make some very successful and critically acclaimed games, including Spelunky, Hyper Light Drifter, and Hotline Miami. It's a proper game engine and the prices reflect this (although you can apply for an educators' discount).

There is a trial version and, while limited, it's enough to give students a taster of what they can achieve. It does require an account to download, but you don't need to attach a payment card to the account to install the basic trial.

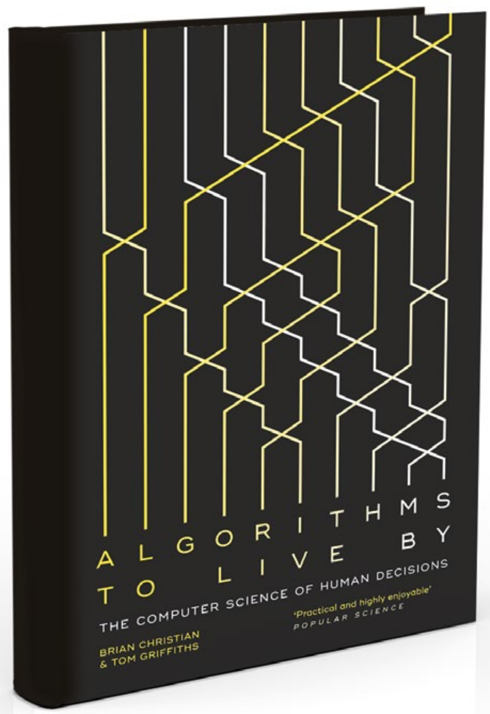
It works surprisingly like Scratch. You can create sprite animations and edit them in almost exactly the same way. You can write scripts for events and movement in text-based code, but there is also a block function that has all the essentials you'll need.

There are plenty of resources, even for the trial version, so it's easy to learn how it works. You can also download examples, see how they were made, and apply that to your own work. Considering the games that have been made with it, it's definitely a very powerful app, and can even be used to publish games for PCs and games consoles.

Gamemaker Studio is much more suited to older students, and does require some teaching to get used to. It is a great way to explore being creative with real, practical code. (HW)



Hello World RECOMMENDS SNAP! #1



ALGORITHMS TO LIVE BY

How computer science can help you solve everyday real-life problems

INFO BY Brian Christian & Tom Griffiths | PUBLISHER William Collins | PRICE £9.99 | ISBN 978-0-00-754799-9 | URL algorithmstoliveby.com

A better title for this book might have been 'Computer Science and Probabilistic Techniques to Live By'. There are some explicit algorithms (72 pages of endnotes and bibliography point you to more) in the Sorting chapter, where big-O notation is introduced, but most of the book is concerned with describing ways of approaching problems, e.g., the Relaxation chapter suggests relaxing some of the constraints on your problem: you might not get the 'correct' answer, but at least you'll get an answer that you can build on.

Optimal Stopping shows how to decide when to stop looking (for a better offer on your house, or a parking place closer to work), while Explore/Exploit examines the tension between getting more information (exploring) or using the information you already have (exploit), and applies this to one-armed bandits, medical trials, why babies explore, and why the elderly reduce their social circle.

Caching, Scheduling, Bayes' Rule, Overfitting, and Randomness chapters all have things to say about how we can apply the techniques suggested by the chapter titles to problems in our lives.

When more memory can be worse

The Networking chapter has fascinating anecdotes about networks and the Internet, a very good introduction to packet switching (not only the 'so it can survive a nuclear attack' approach), wireless networking in Hawaii using ALOHAnet, applications of one of the internet's core congestion algorithms to drug offender supervision, and explanation of why more memory in network devices can sometimes be the worst thing you can have.

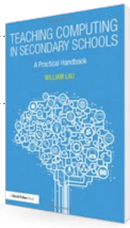
A final chapter on Game Theory brings selfishness and honesty into the realm of algorithms, along with the advice to adopt strategies that don't require 'anticipating, predicting, reading into, or changing course because of the tactics of others', which can not only be easy, but sometimes optimal and, if that doesn't work, then 'try to change the game.'

The Conclusion not only sums up the main themes of the book, but introduces Computational Kindness: when scheduling interviews for the book, the authors found interviewees more likely to be available at, say, 'next Tuesday between 1.00 and 2.00pm PST' than 'at a convenient time this coming week', and suggest that giving a constrained problem ('Can you be free at that time?'), rather than a wide-open one ('When can you be free that week?') was kinder, since it reduced their computational load.

So, the next time you and a group of friends are trying to work out where to go for a meal, saying 'Well, what about the new pizza place?' isn't being pushy, but saving your friends a lot of computation.

Overall this is a fascinating book, perhaps mistitled, but well worth reading for its insights into the application of techniques from our field into everyday life. (HW)

TEACHING COMPUTING IN SECONDARY SCHOOLS: A PRACTICAL HANDBOOK

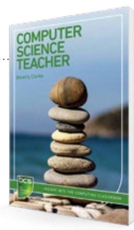


INFO **BY** William Lau | **PUBLISHER** David Fulton / Routledge
PRICE £19.99 | **ISBN** 9781138238060 | **URL** **routledge.com**

William Lau trained to teach through Teach First and is now a CAS Master Teacher and head of Computing at Central Foundation Boys' School in London. He won one of the 2017 Teaching Excellence awards from Infosys Foundation USA, CSTA, and ACM. William's book provides a practical guide to the craft of computing teaching, although much of the book would apply to teaching any subject in an effective, and evidence-informed way.

It's impossible to read this book and not be impressed by the breadth and rigour of William's reading. He has a clearly articulated vision for computing education, and for effective, high impact teaching. He is not afraid to take a critical stance on some of the beliefs common to many classrooms, replacing these with practical, well-thought through guidance on what does work, invariably backing this up with evidence. Not all teachers of computing would necessarily agree with all that William recommends, but all would benefit from thinking as critically as he evidently has about the issues he raises. (HW)

COMPUTER SCIENCE TEACHER



INFO **BY** Beverly Clarke | **PUBLISHER** BCS
PRICE £19.99 | **ISBN** 9781780173948 | **URL** **shop.bcs.org**

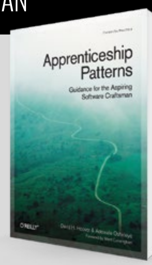
Beverly Clarke is an experienced secondary teacher, and trainer of teachers through her work coordinating CAS's SW regional centre at the University of Plymouth. She has drawn deeply on this experience to produce this accessible guide to teaching computing. Whilst the target audience is those teaching, or training to teach, computing in English secondary schools, we're sure that much of the book would be of interest to teachers of computing in other countries, or indeed in primary and elementary schools anywhere.

Beverly's book takes a pragmatic approach to the realities of computing education, discussing the changing role of the teacher, the technologies and tools available in the classroom, and what career progression might look like. Beverly presents some engaging case studies of typical computing departments, and a useful set of examples and templates. The book reflects the current policy climate of English education, and particularly computing education, well, and Beverly provides ample pointers to policies, reports, and resources. There's limited use made here of academic research into computing, although this too perhaps reflects the broadly pragmatic stance of those currently teaching computing. (HW)

ESSENTIAL READING:

Pattern languages – good solutions to common problems.

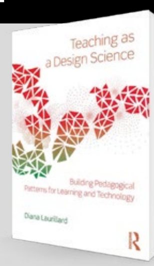
APPRENTICESHIP PATTERNS: GUIDANCE FOR THE ASPIRING SOFTWARE CRAFTSMAN



BY Sophie Deen and her friends
PUBLISHER O'Reilly
PRICE £23.99
ISBN 9780596518387
URL **chimera.labs.oreilly.com**

A clever idea: take the notion of a pattern language, of good solutions to common problems, and apply it to the process of becoming an agile software developer. The patterns here though don't just work for software engineering: most would also apply to those developing their craft as a teacher.

TEACHING AS A DESIGN SCIENCE



BY Diana Laurillard
PUBLISHER Routledge
PRICE £28.99
ISBN 9780415803878
URL **routledge.com**

Laurillard argues persuasively that teaching isn't an art but a design science, with an imperative to make the world a better place. Teachers thus have an obligation to keep improving their practice, test what they do, build on the work of others, and share the design of their work with their peers.

DESIGN PATTERNS: ELEMENTS OF REUSABLE OBJECT-ORIENTED SOFTWARE

BY Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides
PUBLISHER Addison Wesley
PRICE £24.30
ISBN 9780201633610
URL **books.google.co.uk**

A classic text for object-oriented programming, presenting a number of code patterns, each of which addresses a particular, commonly encountered, problem. The authors, the 'Gang of Four', show sample code for each pattern, give some of its known uses and show how it relates to the other patterns they discuss.

EVENTFUL INSPIRATION: PART 3

Our guide to running a successful tech-based event, packed with advice from people who are already running theirs

Are you considering running a hackathon, techmeet or Jam? Are you keen to get like-minded people together to encourage digital making, coding, and community? In issue 3 of Hello World, we took an in-depth look at the activities you could run for your attendees. In this issue, we consider health and safety, safeguarding, and money.

This series of articles is based on The Raspberry Pi Foundation's guide to running a community event. The Raspberry Jam Guidebook is aimed at Jam organisers, but the advice it offers will be helpful to anyone setting out to run a tech-based community event. Ben Nuttall, Community Manager at the Raspberry Pi Foundation, compiled the guidebook. He collected advice from existing Jam organisers throughout the UK, on everything from finding a venue to managing your finances, and from planning your activities to managing social media. Packed full of great, first-hand advice, the guidebook is designed to help you to run the best event you possibly can.

WHAT IS A RASPBERRY JAM?

Raspberry Jams are independently organised community events where people get together to share knowledge, learn new things, and meet other Raspberry Pi enthusiasts. Jams provide opportunities for people to get involved in digital making, develop their abilities, get together, have fun, and socialise. They are usually free or very cheap to attend.

Running a safe event

It is important to make sure that your event is safe to attend for everyone, paying particular attention to the needs of people under 18, and vulnerable adults. It is your responsibility, as the event organiser, to make sure you are familiar with the legislation governing your region, and that you adhere to it in all respects.

It is a good idea to point out the fire exits, toilets, catering, and other facilities while you are welcoming people to the event. If you're not giving an opening talk, ask your check-in volunteers to point these out as people arrive.

Your venue may ask you to complete a risk assessment. Some venues provide examples and templates to help you with this task.

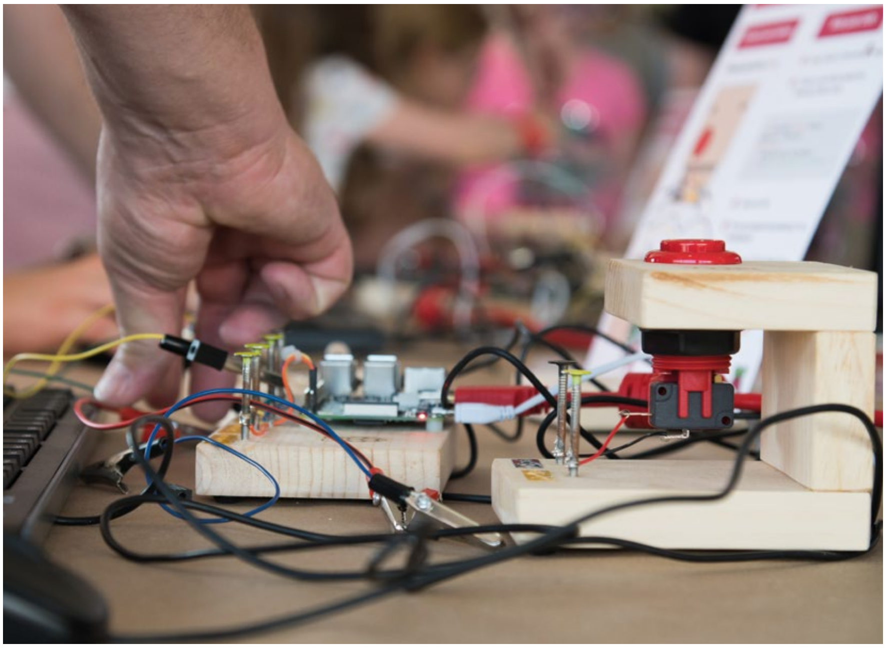
If you are planning activities that involve increased risk to participants, such as soldering, you will need to think about safety precautions. You should plan to provide warnings, explain the dangers, and give sufficient training to anyone wishing to take part.

Don't forget that your participants may bring their own projects to work on or to demonstrate for other attendees. Use your risk assessment to develop a plan for checking their equipment and ensuring that their activities are safe.

START EVERY EVENT BY EXPLAINING WHERE THE FIRE EXITS ARE, AND WHERE PEOPLE SHOULD ASSEMBLE IN THE EVENT OF A FIRE ALARM. TALK TO YOUR VOLUNTEERS AND DISCUSS HOW YOU WOULD HELP A WHEELCHAIR USER EVACUATE. SOME BUILDINGS HAVE A WHEELCHAIR REFUGE AREA WITH A COMMUNICATION PANEL: MAKE SURE YOU KNOW WHERE IT IS.

ANDREW OAKLEY

COTSWOLD JAM



Risk assessment

Unless your venue or local laws insist on it, your risk assessment does not need to be a formal document. Aim to put together a common-sense approach to assessing situations, ensuring that your activities are not putting people in danger, and making sensible decisions to minimise risk.

The Resource Centre website (**helloworld.cc/2tWmEwB**) is a great place to pick up information on running community events. The advice is specific to the UK so, if you are organising an event elsewhere in the world, make sure you are familiar with local rules and requirements. You'll find plenty of resources to help with compiling risk assessments, including templates and advice, such as:

'Risk assessment is about achieving a balance between a reasonable level of risk, and being able to get on with organising your activities. Remember, no activity is completely free from risk, and doing a risk assessment is not about making your activities risk-free.'

Community events should be creative, fun, and safe places for adults and children to learn, and to meet other makers and coders. As the event organiser, you should take reasonable precautions to ensure that all attendees and staff are kept safe. You should ensure that children attending your event are not left unsupervised, and that parents stay with their children.

OUR VENUE, A UNIVERSITY, REQUIRED US TO PERFORM A RISK ASSESSMENT AND HAVE PUBLIC LIABILITY INSURANCE. I HAD EXPERIENCE WITH RISK ASSESSMENTS FROM WORK, SO THAT WAS STRAIGHTFORWARD, BUT ANYONE WHO DOESN'T HAVE THAT EXPERIENCE CAN FIND PLENTY OF EXAMPLES ONLINE. ALTERNATIVELY, ASK SOMEONE AT YOUR VENUE TO HELP YOU FILL IT OUT: IT SHOULD ONLY TAKE TWENTY MINUTES. YOU JUST NEED TO THINK OF THE TOP TEN 'WHAT IFS' AND GIVE SENSIBLE SOLUTIONS. WE BOUGHT PUBLIC LIABILITY INSURANCE THROUGH EVENTS-INSURANCE.CO.UK. REMEMBER TO BUY IT ONLY FOR THE DAYS OF YOUR EVENTS, NOT ALL YEAR ROUND!

ANDREW OAKLEY

COTSWOLD JAM

WE WERE ASKED TO DO A RISK ASSESSMENT FOR THE INSTITUTE OF ASTRONOMY WHEN WE WANTED TO START INTRODUCING SOLDERING WORKSHOPS. WE NEEDED TO THINK CAREFULLY ABOUT ALL THE POTENTIAL RISKS AND WORK OUT HOW WE MITIGATE THEM. IN THE CASE OF SOLDERING, WE NEEDED A BUCKET OF WATER FOR BURNS AND A FIRST AIDER ON HAND, JUST IN CASE.

MICHAEL HORNE

CAMBRIDGE RASPBERRY JAM

I HOLD A MONTHLY JAM AT THE RASPBERRY PI OFFICES IN CAMBRIDGE, WHERE WE HAVE A PERFECTLY SUITABLE SPACE TO RUN AN EVENT. I DISCUSSED IT WITH OUR OFFICE MANAGER AND OPERATIONS DIRECTOR, WHO HELPED ME CALCULATE THE MAXIMUM CAPACITY OF THE SPACE, AND GO OVER ANY PRECAUTIONS OR RISKS WE SHOULD CONSIDER. THEY WERE REALLY HELPFUL AND MADE IT SO MUCH EASIER TO PLAN MY FIRST EVENT.

BEN NUTTALL

RASPBERRY JAM @ PI TOWERS

Safeguarding

You must consider the safety of under-18s and vulnerable adults at your event. If your event is taking place outside the UK, make sure you are familiar with local laws and regulations. Make it clear that any adults at your event must:

- Never be left alone with young people or vulnerable adults
- Take care not to have physical contact with young people or vulnerable adults attending the event
- Not take or share photos of others without appropriate permission

Your venue may have an existing safeguarding policy. Ask to see it, and follow any guidelines it provides.

ALTHOUGH MOST OF OUR VOLUNTEERS ARE DBS-CLEARED THROUGH OTHER VOLUNTARY WORK, WE MADE A CONSCIOUS DECISION THAT WE WANTED PARENTS TO REMAIN WITH THEIR CHILDREN. THE RESPONSIBLE ADULT DOES NOT NEED THEIR OWN TICKET, BUT REQUIRING PARENTS OR GUARDIANS TO ACCOMPANY THEIR CHILDREN REMOVES THE NEED FOR DBS PAPERWORK, AND MEANS THAT EVERY CHILD HAS A FAMILIAR AUTHORITY FIGURE ON HAND. WE ALSO FIND THAT CHILDREN ARE MORE FOCUSED IN THE WORKSHOPS WHEN THEIR PARENTS ARE AROUND.

ANDREW OAKLEY

COTSWOLD JAM

FINANCIAL MANAGEMENT CASE STUDIES

Online Application

A bank account is something we have been looking into for a while. We looked around the major banks and shortlisted it down to two. These banks both offer free community bank accounts that are basically business bank accounts, free of charge for not-for-profit community organisations. We ended up choosing the bank whose application process could be completed online. The process involved filling out a long web-based application form, covering everything from the purpose of your organisation and uploading a formal rules and regulations document, to your tax status, and so on. For any section we weren't sure about, we simply gave them a call and queried it. The entire process took about a month, with the result being a full business bank account with debit cards and a chequebook, and the option to require everything to be countersigned. This is certainly a time-consuming process if your event isn't an official organisation, but being able to manage the event's money from a central location is something that will pay off in the long term, especially if there are ever any questions asked about where donations and funds have been spent.

ANDREW MULHOLLAND

NORTHERN IRELAND RASPBERRY JAM

Loyal customer

Between the two of us, we looked at our own personal banks to see what options were available. We decided that, seeing as I do most of the paperwork, we should go with my bank. They required lots of forms of identification and other paperwork, as well as a personal meeting with a banker. It took them a while to sort everything out, but we're much happier now that the money goes directly into that account. It helps keep us organised!

MICHAEL HORNE

CAMBRIDGE RASPBERRY JAM

Handing over

The venue I used for the Manchester Jam was a community space which held other tech meetups. Since we used the whole venue for a whole day, I offered to donate all the ticket money to them, to put towards their running costs. Any extra donations or sponsorship we got, I put towards buying pizza at the event, and any surplus I saved, keeping track of how much we had. When it built up to about £200, I decided to buy some Raspberry Pis for the Jam. When I moved away from Manchester, I agreed to hand over the event organisation to one of my volunteers, and I left him with the remaining balance.

BEN NUTTALL

MANCHESTER RASPBERRY JAM

Financial management

It is your responsibility, as the event organiser, to make sure you are familiar with any regulations governing the management of funds for community groups in your region, and that you adhere to them in all respects. The advice in this section has been put together with reference to UK-based events.

You may find yourself holding funds after your event, so make sure you keep a careful record of income and outgoings. This is particularly important if you are holding event funds in a personal bank account. Use a spreadsheet to identify how much money belongs to the event, and make sure you don't end up spending your own money. The Resource Centre website offers advice on taking care of finances on behalf of a community group (**helloworld.cc/2sJc0Ks**).

If your event ends up generating a lot of revenue, think about keeping the event money separate from your private accounts. If your event is based at a school, university, library, or community space, you could ask them to hold the money for you, and make payments and purchases on your behalf. Alternatively, you could set up a community bank account. Some banks offer special accounts for small voluntary and community organisations, which provide free banking, as long as your account is in credit.

Make sure you can easily transfer ownership of the account, money, and any equipment you own to someone new if you decide to hand over the running of your event. Using bank accounts with multiple signatories, and keeping careful records of the ownership of equipment, will help to make a handover more straightforward. If you cannot find anyone to take over the running of your event, we suggest that you pass on any assets to another, similar event in your area.

Local technology firms are likely to want to encourage the teaching of digital skills in the local area, and you may find that they are willing to sponsor your event as part of their corporate social responsibility and community outreach initiatives.

The best way to approach a company for sponsorship is to specify an aspect of the event to be supported by their funds. You can't expect them to cover all your costs, but if you have something in mind that they can contribute to, this will help. Good examples include buying pizza for the event, or buying a set of Raspberry Pis for workshops.

Be grateful for any donations or sponsorship you receive, and be sure to thank the donors privately and publicly. A 'thank you' tweet from your event's account will be well-received, and companies will appreciate the publicity. Encourage attendees to retweet, and to add their own thanks to your message.

As well as local companies, you could approach Raspberry Pi retailers to ask whether they would sponsor an event. If you can give them space to promote or sell their products, they may find sponsorship to be a very worthwhile gesture. Note that if you are planning to allow companies to sell products at the event, you should check this with your venue first.

If you ask a company for sponsorship and they say no, be polite and move on. You may approach a company that has sponsored other events, and find that they refuse when you ask. Don't take offence, as this could simply mean that they have spent their sponsorship budget for the year. (HW)

> IF YOUR JAM RUNS REGULARLY, IT PROVIDES BOTH AN INCENTIVE AND AN OPPORTUNITY TO RAISE FUNDS

Fundraising

If your Jam runs regularly, it provides both an incentive and an opportunity to raise funds. If you take in more than you need, you can keep the remainder as a float for the Jam. You can use this to cover future costs, or save it to buy equipment for the Jam.

If you find that your Jam would benefit from buying some equipment, and your leftover ticket money isn't enough to cover what you need, you might want to consider doing some fundraising. A simple option is to ask for donations – using a 'donation' ticket type, or even having a donation tin at the event. Alternatively, you could run a crowdfunding campaign to raise awareness of your group's community activities, and ask for donations to provide more opportunities to local kids.

You could even organise an old-fashioned fundraiser, like a sponsored walk or a cake sale! If you can get all your members involved, they can help to promote the community message.

Sponsorship

If you are aiming to raise funds for your event, why not contact companies and organisations to ask for sponsorship?

DOWNLOAD

Download your copy of the Raspberry Jam Guidebook from: **helloworld.cc/2p7ZX98**

FUNDRAISING CASE STUDIES

Raffle

In order to avoid charging for tickets, we always hold a raffle and raise between £60 and £200, and everyone's happy whether they win or not. You can use Twitter to ask for swag from the community, to give away as prizes. It's so nice to be able to run community events which are free to attend.

GRACE OWOLADE-COOMBES

SOUTH LONDON RASPBERRY JAM

We hand out free raffle tickets while we shake the donation tin. Then we pull winning tickets from a box and they win a prize we've had donated, like a HAT or something.

ANDREW OAKLEY

COTSWOLD JAM

SD supplies

We have some SD cards which we loan out for the afternoon, and other cards which we sell at a small profit. If you can find a cheap source of decent blank SD cards, then it's a good way to raise funds while also solving a big problem.

ANDREW OAKLEY

COTSWOLD JAM

Crowdfunding

Initially, my son Femi ran a highly successful crowdfunding campaign using Indiegogo, by producing a one-minute film about what he wanted to do and the equipment he would need. We set the target at £500 over a two-week period. We got that in four days, and went on to raise £1,270 which has lasted us for 18 months. We went to all the Jams we could, and Femi did talks at places like CamJam and PyCon. We've had donations of kit from Raspberry Pi retailers.

It's really important not to overdo the amount you aim to collect. The key thing is to meet people in person, build up a relationship, and always send thanks via Twitter and blog posts. You really shouldn't automatically expect companies to donate supplies to you, but be really grateful when they do.

GRACE OWOLADE-COOMBES

SOUTH LONDON RASPBERRY JAM

YOUR LETTERS

Our letters page is a place for you to join our conversation. If you've got a comment, a question or an announcement to share, contact us on Twitter via **@helloWorld_Edu** or using the **#HelloWorld** hashtag. Alternatively, email us with 'Teacher Letter' in the subject line (**contact@helloworld.cc**).

Vector Grid components in Scratch

Dear Hello World,
I am an Irish member of the CAS Community (where I have shared resources) and I have begun an online discussion.

On 10 June, the research team at Scratch MIT added new vector grid components into Scratch. To see examples; go to scratch.mit.edu and type 'vector grids' in the search box. As a result anyone, (children, students, teachers, any adult), can now use Scratch in ways not previously possible, especially in school. This MIT decision followed a short period of discussion between myself (Seamus O'Neill, a retired Irish teacher) and Natalie Rusk, Dir. of Learning Resources and Research MIT (Boston). Scratch is now much better fit-for-purpose as a methodology for constructivist learning, and a tool for teaching Computational Thinking (CT). The grids are the Magic Mix, to make coding easier to 'get on top of' for teachers, and to free up valuable teaching time.

I have been following the ScratchMaths.org project, which received several million pounds' funding from EEF in 2014. My desire is to make teachers' work easier and more rewarding. For me, since retirement, Scratch Maths R&D (and all subject areas) is an enjoyable pastime (backed up by my volunteering as a Code Club mentor and Teacher Tutoring at all levels.)

Follow up Conference: On 20 June, before schools closed for Summer, I ran the first ever Irish SCRATCH Conference for teachers (primary, secondary, and further education and adult ed.)ages 8-18 years and upwards. The one-day conference consisted of 3 workshops (each with its own six-page laminated book). Feedback was excellent and 100% of the attendance of 60+ were very pleased with the new component of Scratch.

Seamus O'Neill

If you're interested in learning more, you can reach out to Seamus via the website **readysteadycode.ie** or via CAS Community.

Article discussion

Dear Hello World,
While I'm an academic instead of a teacher, I have comments on the latest issue of Hello World. I think the letter section should include discussions of previous articles, not only questions from teachers.

In the FAQ section of issue 3, the first FAQ, "How do I introduce my students to coding on a mobile phone?", suggests ScratchJr as a possibility. I'm nearly certain it only runs on tablets and not on phones. Pocket Code (catrobat.org) is a free app that runs on Android phones, (and an iPhone version is under development), that is a very powerful, yet easy-to-use, programming tool whose interface works well on phones. What is particularly exciting about it is how easy it is to use the phone's sensors such as tilt, loudness, compass direction, etc.

Another question is "I need to teach programming, but I'm not allowed to install anything on the school's network. What do I do?" The answer is a very good one and mentions the developer tools of the browsers. Chrome's developer tools provide a complete IDE with local file storage and more. See helloworld.cc/2AP2YxP.

Ken Kahn

Thanks for getting in touch, Ken!

HELLO WORLD SURVEY

Over 1400 Hello World readers recently gave us valuable feedback on how the magazine had performed over the past year. 48% of responders were primary or secondary school teachers, 65% came from the United Kingdom, and 28% were female and 72% male, respectively.

When asked how much of the magazine was being read, 71% said: "at least half the content or more". 65% also said they use content from Hello World "several times a year or more" in their teaching practice, which is great to hear. It's unsurprising then that 'lesson plans' and 'practical examples for other educators' are the types of articles that they found the most useful.

The top three topics they feel most confident to teach are: visual programming languages such as Scratch, computational thinking, and generalist programming concepts. In future issues, readers asked to see more practical examples for computing project ideas and lesson plans.

78% voted to double the amount of issues of Hello World, from three to six, and a reassuring 91% said they'd be okay with relevant advertising being introduced to the magazine.

Thanks to everyone who gave their input! (HW)

MILES BERRY PRINCIPAL LECTURER

TEACHERS' KNOWLEDGE

The three overlapping areas of knowledge explained

It's undoubtedly true that the quality of any education system is dependent on the quality of its teachers, and that the only way to get excellent computer science teaching in every school is to get an excellent teacher of computer science in every school. But what should an excellent teacher of computer science know? And what sorts of initial teacher training, or subsequent professional development, is likely to produce excellent teachers of computer science?

Back in 2006, Mischra and Koehler, building on earlier work by Shulman, developed the 'TPACK' framework: I think this provides one helpful way of attempting to address these questions. Put simply, TPACK recognises that there are three overlapping areas of knowledge which teachers need: technological knowledge, pedagogical knowledge, and content knowledge. Mishra and Koehler go into more detail in considering the interactions and overlaps between these three areas, but let's just think about the three areas in turn here.

Content

I'm at risk of stating the obvious, but in order to teach computer science well, a teacher has to know some computer science. Whilst we might argue that the role of the teacher is now much more 'guide on the side' than 'sage on the stage', there seems to be fairly robust evidence that pupils learn most when their teacher does know what they're talking about. In developing the 'Barefoot Computing' CPD for primary teachers, we started from the premise that teachers already knew how to teach and, by and large, knew how to use technology, but most in primary schools didn't know much, or any, computer science. Similarly the QuickStart guides I wrote for CAS/BCS were focussed quite tightly on giving teachers the subject knowledge they needed to teach primary and lower-secondary computing. There are many excellent resources out there for teachers who want to learn computer science: books are great, as are online interactive tutorials (to a point), as well as many MOOCs, including those from the Raspberry Pi Foundation.

Technology

Whilst many teachers do have a good level of technology skills relevant for their broader professional role, I'm not sure the same is true when it comes to the specifics of programming and digital making. Even coding in Scratch seems a big step up from creating presentations, and text-based coding, robotics, or data science make even bigger demands on teachers' technical skills. Classroom confidence does seem to depend, at least in part, on being a master of the tools of ones trade and, for the computer science teacher or digital making educator, these tools are typically quite complex, flexible, and powerful. Papert's great insight, that learning happens through making, applies to teachers as much as to their pupils: mastering the tools of our trade comes through using them productively, creatively and (often) collaboratively: sign up for the next Picademy to see this in action.

Pedagogy

How we teach computing is interesting territory, and I'm not sure that we can take this for granted. Most subjects in school have a long-established body of pedagogic practice: we teach, very often, how we were taught. This doesn't work for computer science: it's such a different subject from ICT that we can't assume what worked there works here, nor can those of us with a degree in CS simply take the practice from higher education and apply it in school. For me, this is what makes the subject so exciting: that teachers are by and large, figuring out for themselves what works: generously sharing their insights with their peers through CAS hubs or CSTA chapters, as well as experimenting with innovative approaches and writing up their findings, as in the BCS Certificate scheme.

All of this, of course, takes time. Great as Barefoot, Picademy, the BCS Certificate and all the other initiatives are, none will work if teachers don't have the time to participate in these, integrate these different forms of knowledge, and try what they've been taught out for themselves. (HW)

Miles is principal lecturer in computing education at the University of Roehampton. He is a member of the CAS and CSTA boards and the Raspberry Pi Foundation.

REFERENCES

Mishra, P. and Koehler, M.J., 2006. Technological pedagogical content knowledge: A framework for teacher knowledge. Teachers college record, 108(6), p.1017.

Shulman, L. S. (1986). Those who understand: Knowledge growth in teaching. Educational Researcher, 15(2), 4-14

"HELLO, WORLD!"

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

Q WHAT IS HELLO WORLD?

A Hello World magazine is a new 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). Hello World is sponsored by BT.

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 inside and outside the classroom.

Q WHEN IS IT AVAILABLE?

A Your new 100-page magazine will be available three times per year in time for each new term in January, April, and September. Would you like it to be available more frequently? Let us know!

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 for a FAQ section 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? We'd like to share it.
- **Write for the magazine**
  Do you have an interesting article idea? 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

fb.com/HelloWorldEduMag

SUBSCRIBE IN PRINT TODAY!
PAGES 28-29

(Hello World)

helloworld.cc