

# COMPUTING AT SCHOOL

EDUCATE · ENGAGE · ENCOURAGE

In collaboration with BCS, The Chartered Institute for IT

## SWITCHED ON

COMPUTING AT SCHOOL NEWSLETTER

SUMMER 2013



# THERE IS NO 'THEM' ONLY US!

These are exciting times for CAS. In January, Michael Gove announced that Computer Science would count towards the English Baccalaureate, placing the subject as “the fourth science”. In February, the new draft Programme of Study re-titled the subject as “Computing”, and explicitly establishes Computer Science as a deep subject every child should have the opportunity to learn from Key Stage 1 onwards, alongside the creative use and application of information technology. These are goals for which we scarcely dared hope four years ago.



CAS Chair, Simon Peyton-Jones

Now we have to turn the dry bones of a programme of study into an inspirational subject discipline in every school, for every child. That is a huge challenge, and one that will be met largely by the professionalism and creativity of our existing ICT teachers. In April, CAS membership passed the 4,000 mark. That same month we heard the DfE had made £2 million available to fund an extension of the Network of Excellence. We hope to recruit more Master Teachers and strengthen the collaborative work with University Computer Science departments to further build

capability in the classroom. The CAS Community already has nearly 700 resources posted by teachers willing to share their ideas. Whilst there is an active forum, it's the resources that give a real sense of teacher's willingness to 'walk the walk'. This grassroots activity is central to our work. There is no them, only us! As CAS Chair, Simon Peyton-Jones recently commented. “The DfE funding will be incredibly helpful to oil the wheels, and provide some coordination, *but the real motive power remains, well, us*: teachers, IT professionals, software developers, awarding bodies, employers, and so on. And I don't mean “us, CAS” either! I mean “us, CAS, Raspberry Pi, Naace, ITTE, MirandaNet, Code Club, YOUSRC, YRS, Hack to the Future, Technocamps, etc”. Now more than ever, we need to work together.”

## INSIDE THIS ISSUE

The energy of CAS members knows no bounds. There's so much going on that this issue is bigger than ever before. Events of the last few months have fired the 'can do' attitude of teachers. Inside you'll find articles by teachers from all over the UK, suggesting resources you can use straight out of the box to get your pupils going.

Rik Cross and Dan Fraser share the resources on their wonderful websites. Lucy Bunce points you towards a book to introduce computing concepts to any age. Mark Dorling shares another fantastic lesson idea and Alastair Barker tells how he built interest in his school. Ray Chambers highlights a new resource whilst Chris Swan talks about tinkering with Lego and the Raspberry Pi. Ben Smith and Peter Kemp extol the virtues of Blender whilst John Stout and Lyndsay Hope look at resources of value for older students.

Pedagogy will be key to developing a new Computing curriculum. Greg Michaelson shares his thoughts about approaching Computational Thinking and Peter Donaldson looks at research into teaching programming. There's much more too. We hope you enjoy it, and will be inspired to contribute too. There is a unique spirit of collegiality in CAS. If you haven't already joined, please do.

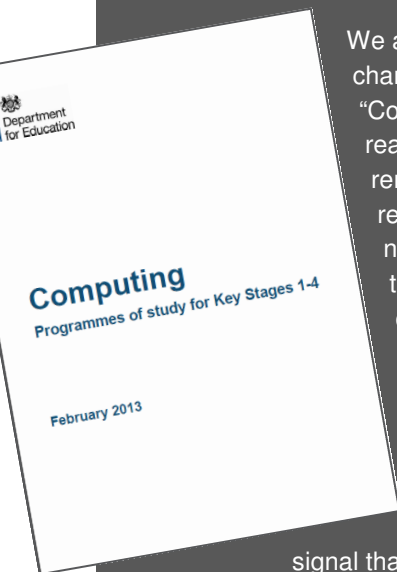
The “Computing At School” group (CAS) is a membership association in partnership with BCS, The Chartered Institute for IT and supported by Microsoft, Google and others. It aims to support and promote the teaching of computing in UK schools.

ISSN: 2050 -1277 (online) 2050 -1269 (print)



## COMPUTING: A NEW SUBJECT FOR THE NATIONAL CURRICULUM

Consultation has recently closed on the draft National Curriculum programme of study. A significant change was the introduction of a new subject Computing, replacing the existing ICT. The proposed programme of study for Computing clearly establishes Computer Science as a deep subject that every child should have the opportunity to learn, from primary school onwards. CAS clearly welcomes this.



We also welcome the change in title to “Computing”, for two reasons. First, it removes the explicit reference to technology, a reference that pulls in the opposite direction to that of trying to establish Computing as a subject **discipline**. Second, it gives a clear

signal that something has changed, and that head teachers, parents, governors, and students, should take notice and act.

The government indicated that the proposed programme of study should be ambitious. It's clear the Department are taking on board many of the proposals made in the recent Royal Society Report “Shutdown or Restart? The way forward for computing in UK schools”. The report describes Computing as encompassing Computer Science, IT and Digital Literacy. The challenge for teachers will be to develop schemes of work that will allow students at the end of KS3 to make informed choices about appropriate qualifications in all these areas. In particular, many teachers will want to know what constitutes Computer Science and how to teach it. We have consistently said there is a lot more to Computer Science than simply programming. At its heart is the notion of Computational Thinking. In forthcoming issues of **SWITCHED ON** a key focus will be to explore this theme and ways to develop the associated pedagogy.

## DFE BACKS CAS NETWORK OF EXCELLENCE FOR TWO YEARS

**In just five months since the launch of the Network of Teaching Excellence in Computer Science (NoE) some 700 hours of CPD have been provided. DfE reviews have been extremely positive about progress made thus far.**

The Network started with £200k of seed-funding from DfE, Microsoft, Google, CPHC, OCR, AQA and BCS. Over 600 schools and 70 universities (including 18 out of the 24 Russell group universities) signed up to be part of the Network in the first six months surpassing our expectations. Over 120 schools have committed to becoming lead schools that support the development of CPD activities for other network schools. An initial cohort of 32 Master Teachers have delivered over 70 different CPD courses for teachers in their local area.



The heart of the programme is to build a high-quality, sustainable CPD infrastructure at low cost. This will be achieved by nurturing long-term, bottom-up collaboration between employers, universities, professional bodies, schools and teachers. We have just received news that the DfE have supported the application made by CAS/BCS to continue and expand the NoE. The funding we have received is for two years but we are working on a five year programme to:

- Recruit 600 CAS Master Teachers (primary and secondary)
- Harness university expertise to lead on training and development of the CAS Master Teachers
- Maintain comprehensive classroom resources for all key stages
- Enhance professional status for Heads of Computing in schools

We are all aware of the short timescale and are looking to extend the existing Master Teacher programme starting in September. See the CAS NoE discussion forum for specific details. If you are interested in getting involved email Mark Dorling ([mark.dorling@computingatschool.org.uk](mailto:mark.dorling@computingatschool.org.uk)).

We haven't got everything right yet, but are confident that the model of teachers supporting others locally with help from local universities is the right way to proceed. Our focus is to establish sufficient Master Teachers, working in partnership with universities, Teaching School alliances and professional bodies, to offer not-for-profit CPD to almost all schools in England. It is an ambitious, but exciting, project and we are looking for your help. We have a once in a generation opportunity to make a real difference in our schools. We can do this by helping and supporting each other. Will you help?

*Simon Humphreys*

# THINKING ABOUT COMPUTATIONAL THINKING: PUTTING INFORMATION BEFORE COMPUTATION

**Computational Thinking (CT) is an approach to problem solving based on concepts of decomposition, pattern recognition, pattern generalisation and abstraction, and algorithm design. It is fundamentally a way of thinking. Greg Michaelson, from Herriot-Watt University suggests ways to introduce the concepts in the classroom.**

CT has an emphasis on understanding the general characteristics of problems independently of programming language or platform. However, to apply CT to concrete problems, it is tempting to seek some formulaic prescription for the sequential application of these concepts. For most problems, though, these concepts can't be deployed in isolation from each other. Indeed, the concrete instances of CT concepts for a problem can often only be teased out in retrospect, once the solution has been determined. Nonetheless, bearing CT concepts in mind gives valuable structure to problem solving as a process of refinement, narrowing choices framed by the concepts towards a unified solution.

Unfortunately, CT seems at odds with the equally contemporary "programming is the new Latin" approach, of teaching all students to program as early as possible. Typically, very small, single purpose programs are constructed from pre-given components, within simple, animated IDEs. This approach is highly motivating, building student confidence in their ability to program.

The trouble is that this approach doesn't scale: students find it hard to make the leap from plugging together small programs to designing even moderately sized ones. But here CT feels irrelevant: given simple enough problems and the right components, solutions seem "intuitive" or "obvious". Thus, systematic programming seems irksome compared with just having fun, making the machine do cool stuff. We can reconcile CT and early programming by revisiting Niklaus Wirth's

succinct 1973 koan that "Algorithms + Data Structures = Programs". Forty years later, we might say Computation + Information = Solutions.

And I think that CT overemphasises Computation at the expense of Information. To go back to basics, problem solving involves characterising how to transform an initial state of being into a final state of being, that is how to change the organisation of the information. Thus, we should think about how the information relevant to a problem is naturally organised. To help us, we should use familiar structures like sequences, tables, records, vectors, grids, trees and graphs.

These are not data structures, which are programming concepts. These are *information structures*; ways of characterising how information is organised. The only commitment is to manipulating the structures through particular operations, to find, update, delete and insert information. In language terms, if information structures are nouns then operations on information structures are verbs, which we eventually use to compose sentences, that is programs.

This may seem far too heavy weight for classroom teaching, even though ex-6<sup>th</sup> formers with no prior programming experience routinely meet such ideas in 1<sup>st</sup> year University courses. Nonetheless, the world is full of really interesting information scenarios: I think we can teach problem solving, and hence programming, right from the start by interrogating concrete problem instances, using ideas from CT to tease out information structures.



As an exercise, consider each of the following scenarios: How might you:

- find your car in a car park?
- find someone's office if you don't know their room number?
- check if the shopping bill confirms that you've bought everything on the shopping list?
- identify the best available seats in a theatre?
- tell if two people are related?
- organise the seating at a wedding reception so as to avoid people who don't get on sitting next to each other?
- make sure you don't give the same present to the same person two years running?
- tell which additional ingredients you need to buy to cook a dish from a recipe?

Try to solve each scenario, by decomposing the problem to find patterns of information access and manipulation which can be abstracted via the familiar information structures mentioned in the main article. Which structures are appropriate? Which don't work? Why? Always "why?"...

This article is digested from the presentation "Computational Thinking is Informational Thinking", available from: [www.youtube.com/watch?v=U9EVTvJvTMw](http://www.youtube.com/watch?v=U9EVTvJvTMw)



## CPD: NEW AND ASPIRING LEADERS OF COMPUTING

Based on successful courses for heads of science and mathematics, the National STEM Centre, in conjunction with CAS, is planning to launch “New and Aspiring Leaders of Computing” CPD provision. It will have generic components including:

- creating a vision for your department
- systems to support learning
- developing an outstanding teaching and learning strategy
- monitoring and evaluation
- coaching
- leading change
- assertiveness skills

The focus will be on effective pedagogy rather than how to “do computing” and will be complementary to the growing spectrum of courses under the Network of Excellence. But what about areas specific to a Head of Computing (which would differentiate the course from its Science and Maths stablemates)? Suggestions so far include:

- Managing change and exploring the range of KS4 and A level qualifications.
- Developing pedagogy to support learners with marked differences in levels of knowledge and ability.
- Links to other subjects.
- Understanding your technical environment and the limits school network access and infrastructure may place on innovation. Engaging and supporting school technicians and network managers.
- Professional networking and building relationships with clusters of schools to share good practice.
- Keeping head teachers informed and raising the profile of computing in your school
- Careers advice and enrichment activities
- Attracting more girls into a discipline in which they are under-represented
- Extra-curricular opportunities: programming clubs, competitions, robotics

If you have any thoughts on the content of such a course please feedback via the [CAS community forum](#). Further details will be available shortly. If you would like to be kept informed please e-mail: [enquiries@nationalstemcentre.org.uk](mailto:enquiries@nationalstemcentre.org.uk)

*Paul Browning*

## INTRODUCING COMPUTING: A SUCCESS STORY FROM DORSET

**How do you introduce Computing? In 2009, Alastair Barker, the Subject Leader for Computing and ICT at St. Edwards School, Poole was given the go-ahead to offer Computing (alongside ICT). Here’s how he did it.**



The main problem was never going to be teaching the theory. It is always a steep learning curve when learning a new scheme of work, but there are lots of resources available to help in this area. The main problem was getting students to be able to program and think computationally. Working on the principle of “little and often” I decided to start early embedding Scratch, RoboMind, Kodu (but mainly Scratch) into KS3 basing teaching around the four principles of variables, iteration, flow control and I/O.

Whilst this was bubbling away lower down, I started the A-level Computing course and phased out the A-level ICT. The A-level group was small at first and we lost some students who didn’t realise what it involved. I quickly realised the first thing I needed to do was to ensure there were no links between Computing and ICT. Students were very disenfranchised with ICT and I wanted them to know that Computing was a different animal. One thing I did (other than to rename the department to “Computing and ICT”) was get a maths teacher to teach a major part of the AS course. He was happy as it helped him learn to teach A-level Maths (D1) and it gave the Computing department a bit of kudos from the well established and respected maths department. Lower down the school I ensured students knew at the start of lessons whether they were to be taught computing or ICT so that at options time they would have an idea of the difference between the two.

In year 3 of the move (2011) a pilot of the Computing GCSE was offered to a select group of year 11 ICT students who had completed their ICT GCSE in year 10. The following year we put GCSE Computing on the options alongside ICT. It was well received. We advertised via the maths department getting them to single out those students who might have a leaning towards computing and pointing them in our direction for further information. Year on year the Computing A-level has become more and more popular aided not least by the current atmosphere—parents would ask about Computing at parents evening. With GCSE Computing now fully up and running it is making the future look very exciting.

Overall I think raising a positive profile of Computing around school is very important, keeping things as “real” as possible. We entered students into the school music contest to perform a “live coding” set. We bought in Arduinos and created devices that could be used around school. We held Lego Mindstorms days for year 9s that had worked well that year and attended local CAS organised ScratchJams and the new Raspberry Pi is definitely in the development plan. My hope is to hold events where members of the industry can come in and ignite/fan the flames of inspiration and wonder about computing and generally make computing look cool, hands on, relevant and accessible.

# PRIMARY SCHOOL PUPILS START CODING THANKS TO CODE CLUB

**Code Club, the grassroots organisation of after-school programming clubs, is going from strength to strength. A lot has changed since our last article and the growth in clubs has been phenomenal. Volunteer Neil Smith provides an update.**

Everything started on a shoestring, with Clare Sutcliffe and Linda Sandvik (the club organisers) trying to keep it all organised in what time they could squeeze out after their day jobs. Thankfully, some sponsors have stepped forward. ARM and Steer are both pledging money, which means Clare can now work full-time on it, as well as supporting the development of additional project material.

The first Code Club hack day was held last December, in concert with the Raspberry Pi Foundation, NESTA, The Open University, and Free Agent. The brief was to create Code Club projects using the Raspberry Pi. All manner of projects were created, including a light-sensitive zebra crossing simulator and a fully-working mini-band. The theme was inspired by the announcement that Google and the Raspberry Pi Foundation were giving away 15000 Raspberry Pi's to schools and other deserving groups, with 3000 being distributed through Code Club. We're working on the details of how Code Clubs can claim their Pi's, so expect an announcement soon.

Finally, HRH the Duke of York is now Code Club's Patron. His support shows that Code Club is being taken seriously, and we're hoping that his influence will

open more doors for us. The average Code Club has about 15 people, with some going up to 60! We can only assume those are places where whole year groups are working through Code Clubs in lessons. There are about 9500 Code Clubbers, with around 7500 children doing their first programming there. Virtually all the children involved stay in the clubs over the whole term. Our survey responses have generally been positive. Not everything is perfect, and we'll be revising some of the material to fix errors and make improvements.

The first two terms' worth of projects use Scratch, and one common request in the survey was for a different programming language. We're already ahead of the game, with a term's worth of projects using HTML5 and associated technologies developed and now being tested. We're also planning a further term's worth of activities in Python.

Despite the enormous success so far, there's still a long way to go. There are around 25000 primary schools in the UK, so Code Club is only in about 2.6% of schools. There's a long way to go to our target of 25% of schools by the end of 2015. If you'd like a Code Club in your school, sign up at <http://www.codeclub.org.uk/> and we'll do our best to get a volunteer for you!

{ CODE CLUB }

Code Club is a network of after-school programming clubs for primary school children in years 5 and 6. Each club is run jointly by someone in industry (who provides the computing expertise) and a teacher (who does crowd control). Code Club puts the two together and provides some projects for the children to work through. At the moment, we've released two terms' worth of projects using Scratch, the graphical programming environment. At the time of writing, and less than a year since the first one began, there are now over 650 Code Clubs in the UK, which means that about 9600 children are getting some experience of programming, 7500 of which have never done any sort of programming before.



CAS Online

Resources / Discussions / Events

## MAKING NEW TEACHERS FEEL WELCOME IN FORUMS

The most common way most of us "crowd source" solutions to problems is via forums. I am a member of several including CAS. On the CAS forum I am rated as loquacious and possibly, I may even become chatty. We aim to maintain this friendly community as the membership grows. Guidelines are simple:

- Speak the truth in love.
- Be scrupulously polite.
- Some know a lot about teaching and some know nothing; similarly for computer science, or for qualifications. The group is a place to share what we know.
- Try not to dominate conversations. Encourage others to speak.
- Don't forward CAS mailing-list messages elsewhere without asking the author

These 'rules' do a great job in encapsulating the culture we want to encourage. Many readers will have experience of forums rendered useless because of the activities of a small minority. We do not want CAS forums drifting in that direction. It is all too easy to do so if we are not careful about our responses.

No question is too simple, small or stupid. In fact we have have a special category for these. We want the CAS forum to be a place where someone can ask any question about something unfamiliar and get thoughtful and helpful answers. In other areas, robust discussion can help synthesise differing views but it helps if contributors pause to think how they may come across, particularly to new members. The CAS community is a precious resource with a wonderful ethos. Please help keep it that way.

Brian Lockwood

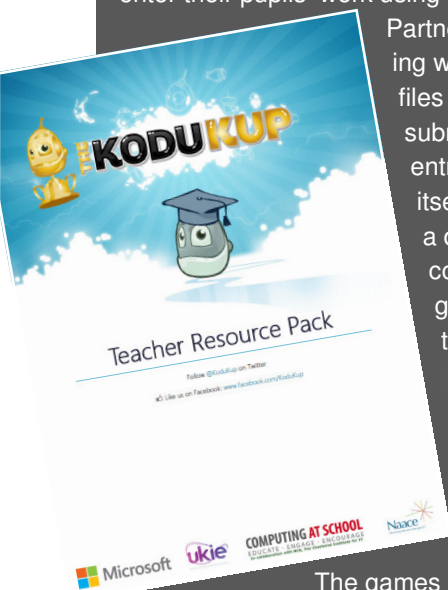


CAS Master Teacher Stacey Jenkins and her pupils demonstrate the dance lesson (right) to Michael Gove

## MICROSOFT LAUNCHES KODU KUP COMPETITION IN UK

The Kodu Kup launched on 30<sup>th</sup> January at BETT and is open to any child enrolled in a UK school who is aged between seven and fourteen. Children may be entered for the competition as individuals or as part of a team of up to three people. You can follow @KoduKup on Twitter or 'Like' us on Facebook ([www.facebook.com/KoduKup](http://www.facebook.com/KoduKup)) to receive regular updates, including dates of free training sessions! Teachers can enter their pupils' work using the Microsoft

Partners in Learning website. Two files need to be submitted per entry; the game itself along with a design for the cover of the game using the template provided. The closing date for entries is Friday 31<sup>st</sup> May.



The games submitted must match one of the following themes: a retro arcade game, raising water awareness or a Mars exploration. Ten shortlisted entrants will be invited to Microsoft Headquarters in Reading, United Kingdom where they will participate in workshops and present their games to a panel of judges. In total, three winning entries will be announced. Each member of the winning entrants will receive an Xbox 360 with Kinect. Amongst the three winning teams, the judges will choose and announce one overall winner of the Kodu Kup. The full details are in the Teachers Pack at <http://sdrv.ms/VHhCSn>

Nicki Maddams

# INTRODUCING COMPUTING CONCEPTS THROUGH DANCE

'Computing through Dance' was developed by The Digital Schoolhouse and Langley Grammar School's ICT Department to appeal to girls and incorporate computing in an innovative way into the curriculum. Mark Dorling explains.

This lesson develops an idea already in use in the Digital Schoolhouse where pupils follow a sequence to re-enact the Michael Jackson Moon Walk as a robot, previously featured in SwitchedOn. The project starts by creating flow charts of instructions to perform dance moves of a well known dance like: the Hokey Cokey, the rugby team (New Zealand And Tonga) Haka, Moon Walk or a Tudor dance which many children study in Key Stage 2. The initial objective is to develop the understanding of a sequence and appreciate the importance of accurate instructions. Loops are then introduced for repeated instructions within the dance. A hilarious video clip of an animated character doing the Hokey Cokey can be used but pupils are always encouraged to volunteer.

Next, the concept of selection is introduced with the introduction of a question; if the whistle blows, then freeze in a pose, else perform the dance. Once the concepts have been understood, students then have to create a dance with four dance moves; the dance must include at least one repeat and a pose for when the whistle blows. The dance sequence is written in a flow chart.



As space can be an issue in any classroom, students are given rules for the dance moves: they have to remain on their chairs and the dance moves are only with the upper body. The class vote on a music style from some given samples. Latin style music always seems to be most popular but the choice of music can always be adapted to suit your curriculum needs. Students can practise some 'hand jiving' to get the idea before their creativity flows. Once the dances have been perfected and the flow charts completed, the students record their dances using video cameras. Peer feedback is next; the videos are watched by the class while the students present their flow charts. A score is given on the clarity of instruction, accuracy of sequence, use of repeats, use of a question and overall quality of dance moves.

The project then evolves into using Scratch. Pupils choose a dance character or import images of themselves to perform a sequence of moves. By building on the students' previous understanding in the kin-aesthetic activity they are able to include repeats and a selection question. The project can be extended to add a variable to determine the number of times a sequence is repeated and later to introduce the concept of procedures for the more complex dance moves.





# LEARNING HOW TO CODE IN A GAME ENVIRONMENT: **CODE AVENGERS**

In 2011 Michael Walmsley was conducting research at the University of Waikato, New Zealand. While tutoring undergraduates, teaching his children to code and talking to teachers he decided to apply his PhD research to teaching computer languages.

My aim was to develop a resource that made learning to code as fun and effective as possible. Software development is such an exciting and rewarding activity. Why shouldn't learning the basics be just as stimulating, even addictive? The result was CodeAvengers - a gamelike environment with interactive online courses that aim to take the pain out of teaching and learning the basics of computer programming and web development. First-hand classroom experience and extensive discussion with teachers was used to refine the courses to develop an effective learning environment that aims to maximise student engagement. During development a number of lessons were learned.

**Problem:** Many students have limited attention spans. **Solution:** Divide courses into small chunks with gradual progression of difficulty and a variety of interesting tasks. Providing a game-like learning environment—with themed lessons, points and badges—is another valuable technique to keep the attention of teenagers who struggle to find time for homework, but always make time for computer games. The CodeAvengers courses consist of 20 to 40 lessons, each with five tasks. Most tasks take less than five minutes to complete and teach no more than one new concept to avoid overwhelming learners. Variety is also important. The CodeAvengers level 1 JavaScript course includes code challenges, bug

hunts, robot missions and quizzes that introduce, practice and review new concepts. In addition, 30 second reward games are provided at the end of particularly taxing lessons.

**Problem:** Limited reading ability is a stumbling block for many learners.

**Solution:** Use short concise sentences, simple grammar and avoid using uncommon words and technical jargon without a glossary. Humour is best avoided especially when writing for an international audience, to avoid confusion. When writing course material, it is important to have the target audience in mind. CodeAvengers task descriptions aim to be easy to read for 12 year olds, and understandable for intermediate-ability, non-native English speakers. Learners can click on difficult words to view explanations. Lessons contain minimal text and complex topics are broken into smaller subtopics that can be explained in a couple of sentences. Concise videos can be useful for introducing complex topics. The downside being that they take more time to create and modify, and are difficult for non-native English speakers without subtitles and an option to slow down playback.

**Problem:** Catering to the needs of all students is difficult. **Solution:** Structure courses so that the majority of students can complete them with little help from the teacher, enabling teachers to focus on those needing the most help. CodeAvengers offers auto-

matic feedback when students make mistakes; hints and answers are also available when students get stuck. It also provides live feedback of class progress for students, indicating how much help students needed to complete tasks. This feedback empowers teachers to quickly identify the individuals who need the most help. Level 1 courses in both html5/css and javascript are free for students to use. Further levels are in development. You'll find more details at [www.codeavengers.com](http://www.codeavengers.com)

## Animation13

The well established and hugely successful 6th Annual UK Schools Computer Animation Competition closed for entries on Friday 22 March. **667** schools registered, and we received **1,120** entries from **154** schools, involving **1,583** students. Teachers who have registered for the competition will be able to download a certificate-of-entry for their pupils in early May.

Prizewinners will be notified by email in the first week of May, and will have guaranteed seats at the Animation13 Awards Festival and Inspirational Computer Science Day to be held at The University of Manchester on Friday 12 July. Full details of the Awards Day and how to register to attend; <http://animation13.cs.manchester.ac.uk/festival/>. Please visit the website to see all the winning entries and photos from last year's Animation12 Awards Day.

Animation14 launches in September 2013!  
*Toby Howard*

Fun effective learning for the total beginner...

# CODE AVENGERS

Learn to code web sites with HTML5/CSS3

Learn computer programming with JavaScript

Level 1 Level 2 Level 3

7-10 hours/level

10-15 hours/level

# #include

computer science for all

8<sup>th</sup> March was international womens day. It was also the launch of CAS #include newsletter. The #include tagline is : 'computing for all'. This is exactly what we aimed to encompass in our first newsletter.

The #include launch party took place on 29<sup>th</sup> April at the BCS offices in London. Attendees were treated to an inspiring speech by BBC Click presenter Kate Russell about how she became interested in technology - beginning when her brother was bought a BBC Micro computer! There was also much interesting debate and discussion around diversity issues.

Coming up is the #define conference: 15<sup>th</sup> June 2013, Rugby School, Warwickshire. The conference is aimed at pupils (11 to 13 years) and

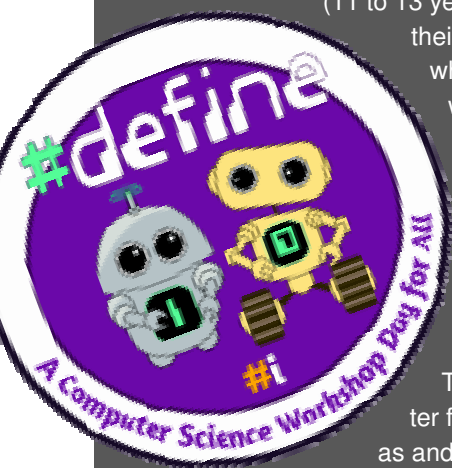
their teachers, where there will be a range of workshops and practical activities to inspire all pupils.

The newsletter featured ideas and resources, such as some posters

from Bruce Ahern, as well as teaching ideas using dance! We also enjoyed Dan Gardner's observations about encouraging girls to study computer science GCSE. These will be available on our website shortly. Check out the newsletter at <http://casinclude.org.uk/maillingMar.html>

#include put a call out for ideas and articles for the newsletter. We got a great response and are really grateful for your input. Our next issue is out summer 2013. In the mean time if you have any resources, lesson ideas or any thoughts on inclusivity that you would like to include in the newsletter. Please email them to [newsletter@casinclude.org.uk](mailto:newsletter@casinclude.org.uk) We do value your input and support.

Reena Pau



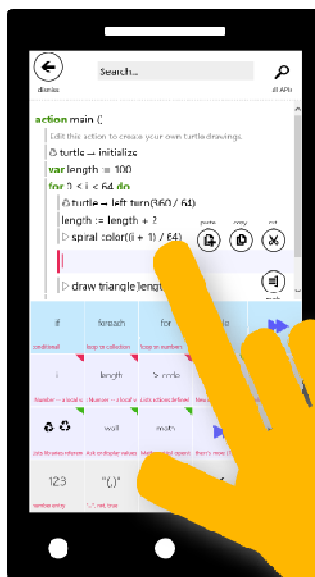
## GETTING PUPILS CODING WITH THE HELP OF TOUCH DEVELOP

**TouchDevelop is a programming environment for both mobile devices and computers. Uppingham Community College teacher Ray Chambers attended a Microsoft 'Appathon' to learn more about its possibilities.**

Any TouchDevelop user can install run, edit and publish scripts. Users are able to share scripts with other people by publishing them through the TouchDevelop interface. You write scripts by assembling on the screen much like Scratch. The apps can be exported to the windows store which will allow students to see the interest they are gathering in the real world. The platform (<https://www.touchdevelop.com/>) is run through a HTML5 compliant browser and allows programmers to build apps "on the go". I have tried it using Chrome, Internet Explorer10 (beta) and other browsers and it works like a charm. The apps themselves are for Windows devices; however the coding can be completed on any device. I have tested it with Apple, Android and Windows mobile devices.

We've just introduced Kodu to our year 7 classes as a step into programming. Our students are starting to understand some of the basics. They have developed small test plans and pseudo code to plan out their games. Our next step is to move to algorithms and logical programming. TouchDevelop fits somewhere in-between Scratch and Kodu and seemed like the next logical approach to take. A developing scheme of work can be found on my blog; <http://www.raychambers.wordpress.com>. The idea is to introduce students to the basics of software development. There are some starter activities to think about the order we use variables and some introductory activities with videos. Having introduced the environment, the first lesson shows them how to use variables to change the background. It also shows how to pass information from variables to control height and width. Further lessons continue to test for interactions such as characters/sprites touching each other.

I'm currently looking at turning the scheme of work where students develop mobile phone applications into a whole school project. It is important to keep students engaged with programming and by giving them some real life scenarios, they can easily do this. They will be approaching members of staff about applications which could benefit their classroom. They will be able to work together to create pseudo code, test plans and explanations about their solution. The code is relatively easy to pick up and you can get making applications straight away. The only thing you will need is either a Windows Live ID, Google account or a Facebook account. These credentials will get you logged in and your applications will be stored in the cloud. This way, your students can access their work on the go. With this sort of availability you may even start thinking about introducing "BYOD" in school.





# THE BEHIND THE SCREEN PROJECT BUILDS A VISION FOR NEW GCSE

**In May 2011, working with Lord Lucas, e-skills UK brought a group of employers, universities and schools together to launch the Behind The Screen project. It is supported by over 50 employers and has 160 schools registered. Sue Nieland explains.**

UK schools can now access the free resources at Behind The Screen: [www.behindthescreen.org.uk](http://www.behindthescreen.org.uk). As the name suggests students need to understand what is 'behind the screen, not on it'. The programme shows students how technology is created and exploited. Ideas to challenge students to think critically and computationally were proposed by employers and turned into e-learning content, with many resources donated by industry.

We started with two projects, focusing on app development (IBM at Wimbledon) and software architecture (The Squawk Project), piloted in February 2012 in ten schools. Both present a brief to students. Each has a Student Log in which research is recorded and activities completed. An exemplar log with assessment information is provided. Sample exam questions, pre-empting the GCSE we hope the project will lead to, are provided. The third project involved Blitz Games Studios and BAFTA. Focused on game design, students able to complete half the project can enter the BAFTA Young Game Designers' Competition. Students completing the entire project develop a game concept emulating those produced in the industry. Following its release, the Department of Business, Innovation and Skills provided us with a grant to look at cyber security. As a result, we released our fourth project in autumn 2012. Skylark Creative designed a 'Cyber City' where students work through a set of seven challenges. Developed again with the cyber security industry, it was shortlisted for a BETT award.

We now have seven projects, with another six in development. Skylark Creative worked with us to create a web design project, where students create a design for a real musician, including mood boards, storyboards and wireframes. Capgemini have co-developed another project focusing on social media and our latest, which was released at Easter, has been produced with AppShed, who very kindly allowed us to include their AppShed Basics Course on our website. Students create an app that helps Year 12/13 students make informed choices between university and an apprenticeship, explaining the financial implications of both.

Future projects include data mining with SAS UK, learning about 'big data' and using industry-standard software (JMP - provided free) to carry out predictive modelling. We are looking at HTML5, CSS and JavaScript coding with Intel and CoderDojo, and coding in Python with the IBM Fish Tank SDK. A team at Intel have already filmed video for us to create a 'Build It and Benchmark It' project to help students understand how computer systems can be assessed for performance. Capgemini have another project in development looking at data centres and their location. As these develop, so is our work on the curriculum for which Behind the Screen is the delivery mechanism. A GCSE with the working title 'Modern Computing' is being developed. As the list of projects suggests, this is a broad and balanced curriculum which includes computer science alongside a range of other topics that employers feel are needed for the future workforce. To find out more or to register to use the resources, visit the website, or email [sue.nieland@e-skills.com](mailto:sue.nieland@e-skills.com).

**mrfraser.org**

## DOING GCSE COMPUTING? MR FRASER CAN HELP YOU

Dan Fraser, Director of ICT at St John's, Marlborough in Wiltshire has made available an evolving collection of resources for Computing at GCSE (OCR), A-Level (OCR) and IB on the website <http://www.mrfraser.org>. The resources include VB.net code tutorials, programming exercises, class notes, presentations, quizzes, exam style questions, past papers, mark schemes and more. Registration is completely free for both students and teachers with a valid school email address (for verification purposes).

The site developed as a result of VLE problems where performance often depended on browser/OS combinations. Dan felt it was important for students to have access to resources and revision materials outside school. What started as a repository for lesson notes and presentations evolved gradually over the next 2 years as he taught himself a new programming language (php) which he hadn't needed to use before.

The resources were originally shared with other ICT coordinators at a regional meeting and word spread from there. There are currently over 300 teachers (some international) registered and using the resources on a regular basis plus an unknown number of students. Most teachers are accessing material for the GCSE computing course as many are struggling when running it for the first time. There are still some gaps (particularly at A-Level) which Dan aims to fill when he finds the time, but already teachers are finding the resources very useful. If you do too, in the spirit of all things CAS, maybe you could also share some of your own.

**Behind  
the Screen** led by  
e-skills uk

## HUBS CAN BRING TEACHERS AND TECHNICIANS TOGETHER

In March I attended my first CAS hub in Lincolnshire. Stories from around the country have intrigued me and left me excited at the move towards Computing. I do not just mean delivering a curriculum but the collaboration of people and their passion to encourage others. With computing becoming more prevalent across many industries (whether it's art, design, astronomy or curing cancer) there is no better time to encourage take up.

Being new to education, my point of view may be unique amongst CAS members. I am an ICT technician in a secondary school. I love computing and wondered if I could instil my passion and interest to help my school. My project started with chatting to staff and management about the infamous Raspberry Pi. Soon discussions turned to how such a cheap device could be used. Why not start with introducing basic electronics, programming and operating systems then move on to designing and making a case for it? This soon led to suggestions to set up a computer club and buy some pi's. I persuaded a friendly ICT teacher and we attended our first CAS meeting. It was really useful. I didn't sleep too well afterwards as a result of the excitement! Perhaps I am a geek? A recent Computer Science graduate from Lincoln University, I also have industry experience.

Do consider those like your Network Manager / technicians who provide you with IT facilities. Maybe we have knowledge, maybe we can help and more importantly - maybe we have the passion! My passion burns deeply everyday. I love to learn, but to pass knowledge on is great. I'm spreading the good work of CAS around and that of the Raspberry Pi, Linux/GNU, Arduino and Makey Makey too. So next time you have a question consider asking your school techie. We can learn a lot from America. The move to free teaching (like CodeAcademy) and educating the masses (some great work from their Universities) can only help the Computing world in general. Hopefully I will meet you at a CAS meeting or event. If I do please introduce yourself, I promise I won't bash you (terrible pun). *Andrew Walmsley*

# BUILDING DIFFERENTIATION INTO CODING CHALLENGES

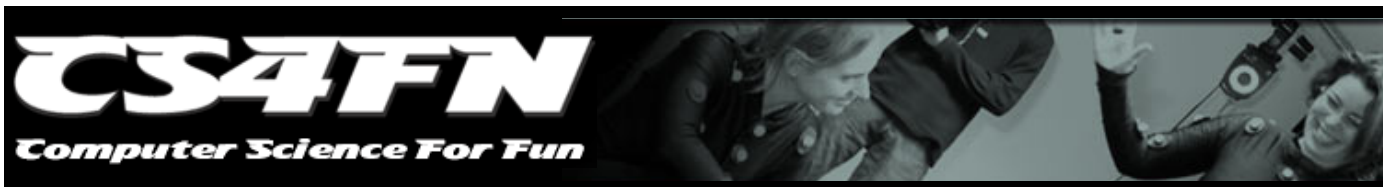
**When a parent approached Rik Cross, a Computing teacher at Roundhay School in Leeds, and asked 'My son can already program in Python ... will he be challenged on your course?' he knew differentiation would be key.**

I had my first programming experience when I was 12, when my uncle gave me his old Commodore Vic-20 computer. It came with a few games, and a book of BASIC code. For some unknown reason, I really wanted to write one of the programs, which made a ball bounce around the screen. I dutifully copied the code listing in the book, and a couple of hours later typed RUN and saw... an error message. I didn't understand the message, so I just checked what I'd typed against what was in the book until I got bored, but I couldn't see what I'd done wrong. In those couple of hours I hadn't done anything productive at all (except maybe increase my typing speed), and I still had no idea how to program.

In preparing to deliver GCSE Computing, I wanted a different approach. I needed a way of teaching programming to some students, whilst at the same time giving other students a chance to apply what they already knew. I decided that giving the students "challenges" would be much more appropriate than giving them step-by-step tutorials to follow. I wanted to teach a programming concept to the students (or at least those who didn't already know about it), and then give them the opportunity to demonstrate their understanding by producing a solution to a different, unrelated challenge. I also wanted this challenge to be as open-ended as possible, by allowing students to add "features" to their solutions, using any programming knowledge they'd picked up.

DATA	EXPECTED	ACTUAL
play == 'y'	Game plays again	Game pl
play == 'i'	Game plays again	Game pl
play == 'u'	Game stops	Game st

I chose Python as the programming language for its simplicity and expressiveness. I liked how a student's first program could literally be one line long, without having to say to them "Type in these other 4 lines of code... you don't need to know what they do yet, I'll explain them later". This means that the solution to the first challenge I set (a simple text banner) can be as little as 4 lines long. But they have ownership over those 4 lines, as they've written them independently to solve a problem. In contrast, I've had students working on later challenges (such as a role-playing game) for weeks. Although I provide pointers as to how a basic solution should look, there's no one correct answer. One unplanned side effect of setting challenges in this way is that students have become extremely creative in the methods used to solve a problem. In fact, some of the later challenges I set have evolved from students improving or adding to my challenges, or just using their knowledge to come up with their own programs. My teaching resources are available at [usingpython.com](http://usingpython.com). Using this website, students learn how to design, program and test a solution to a set of challenges. These challenges include a poetry generator, love calculator, interactive chat bot, and a variety of games. Please take a look and let me know how you get on.



**The cs4fn project started in 2005. Since then it has grown into an unrivalled resource for teachers and students. As the strapline 'Where the digital world meets the real world' suggests, articles popularise issues in Computer Science, making them accessible for school students. SwitchedOn talked to the team that makes it happen: Paul Curzon, Peter McOwan and Jonathan Black.**

The cs4fn project started at Queen Mary, University of London in 2005 with a schools careers fair in the East End of London. Two of us, Paul and Peter, had teamed up to run some workshop activities, and brought along some photocopied sheets with stories we'd written about computing research. We wanted to write the sort of stories we would have liked reading as kids. Stories that showed the fun side of computing – the sense of exploration and discovery that is behind the big ideas. The idea was that if the workshop excited them they would have more to read about to extend the inspiration. Later that year, with help from then PhD student Gabriella Kazai, we wrote more, turning those first few photocopied stories into a full-size, full colour magazine. It was to be given out to people who visited Peter's exhibit at the Royal Society Summer Science Exhibition about Sodarace ([www.sodarace.net](http://www.sodarace.net))

that year. That was the first proper issue of cs4fn. Again the idea was to extend our impact beyond the short time people spent at the stall. We also created the website so there was even more for people to read about if enthused by the magazine. By 2008 we were looking to expand, and received a grant that allowed us to hire someone full-time, which is how we got Jonathan. So now we are a trio!

**Do you have a specific brief? Do you have sponsors that make it all possible?**

Our aim is to use research stories to enthuse people about interdisciplinary

computer science and show that computing is a fun subject. The big grant that has kept us going since 2008 comes from the Engineering and Physical Sciences Research Council. We simply would not have been able to produce and give away so many resources without them. Sadly that grant is ending soon, so the future is a little uncertain. We are looking for ways to make sure cs4fn is still around in the years to come, especially with the new curriculum. Google have also supported us since 2008, helping with the magazine costs



and paying for us to travel to schools and science festivals to excite students in person. We are lucky that both our main sponsors want a very simple thing – more young people interested in computing. They let us get on with doing the job in the best way we know how.

**The next magazine will be Issue 16. Do you have any statistics for the growth in your magazine circulation? Similarly, the popularity of the website must be reflected in the viewing figures.**

Our first magazine printing was only a few thousand copies, but for our latest issue we printed and sent out 31,000 free copies to teachers, students and subscribers in over 80 countries. About 14,000 people visit our website every month, and tens of thousands have now downloaded our magazines and magic books in PDF form. See the boxes right and overleaf for more details of where to find the material.

**Tell us a little bit about the relationship between the magazine and the website.**

You'll find everything that has ever appeared in the cs4fn magazine on our website ([www.cs4fn.org](http://www.cs4fn.org)). But there is a lot more besides. More stories go on the web than we have room for in the magazine. There are also teacher resources and activities at [www.cs4fn.org/teachers](http://www.cs4fn.org/teachers).

We hope teachers will be able to use short features to support their lessons, possibly as homework or prior reading to set some context. We're developing an index on the teacher's page that provides links to articles by syllabus topics. We also provide some hands-on, tried and tested kinaesthetic activities which we've run in many schools. Many have been developed as a result of our outreach work in schools. If you try any of these activities please let us know how it went by using the 'contact us' page.

We publish web-exclusive stories throughout the year on fun, current computer science, so keep checking the 'last one in' section. We also have a major area of the website on women in computing [www.cs4fn.org/women](http://www.cs4fn.org/women) and produced a special booklet "The women are here" to go with it.







Paul Curzon

**Paul, some time ago you wrote a wonderful introduction to computing which is now available for download. Who is it aimed at?**

Thanks! It's called Computing Without Computers, and it's a free book about the core ideas of

computing for total newcomers to the subject. The original idea was to write about programming the way I teach it – focussing on the core concepts rather than worrying about the syntax of languages. I wanted to explain the ideas in a friendly way, without the technical details of real computers.

It's based on the idea that many concepts are similar to things we are familiar with in (non computing) life. These can be used as scaffolding around which to pin understanding of otherwise alien ideas in a memorable way. The book is full of examples that use piles of chairs, elves, mazes, plumbing and Narnia to get across concepts from programming, algorithms and data structures. It can be found by following the links at [www.cs4fn.org/teachers/](http://www.cs4fn.org/teachers/).

In fact for me it was the immediate precursor to cs4fn. Rather than writing new ideas in to it, I started putting them into cs4fn. As the book is full of things I've used successfully in lectures, I hope it will be useful to teachers to help give them ideas. I know that lots of students with little previous computing background have told me they found it invaluable to help them understand what programming is really all about.

**The website is a treasure trove of interesting material. There are simply masses of articles and activities now online. If you were a visitor for the first time, where would you start? Do you have any suggestions for time pressed teachers about finding material?**

A good section to start with might be "The FUNdamentals of computing", where we go through some of the core ideas of computing in a fun way. That's at [www.cs4fn.org/fundamentals](http://www.cs4fn.org/fundamentals). You can also browse all our content by topic at [www.cs4fn.org/sitemap.html](http://www.cs4fn.org/sitemap.html). Or you could look through our latest articles by starting at [www.cs4fn.org/lastonein](http://www.cs4fn.org/lastonein). If you're on Twitter, we are @cs4fn. We tweet links to all our new content online, so you'll never miss anything.

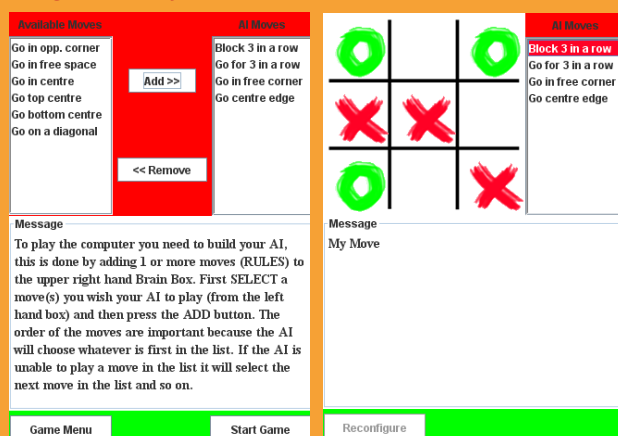
We have been developing an area that includes links to articles that serve as further reading around and beyond the school syllabus too: [www.cs4fn.org/teachers/syllabus/](http://www.cs4fn.org/teachers/syllabus/). Whilst cs4fn articles focus around research we try to teach about core computing concepts in the process, so we do cover a lot of topics teachers would recognise as being on the curriculum. We are actively in the process of adding to this.

**With a greater focus on Computing in schools and many teachers new to the ideas, it's very important to counter the view that computer science is simply learning to program. Could you suggest resources for teachers to get across broader concepts?**

Learning to program is great as long as it helps you dig deeper into what computation is all about. The big rewards come when you start to think about how the ideas fit together. Suddenly one concept might help you solve a problem that seemed totally unrelated. You often hear this sort of thing called 'computational thinking' and it's that, rather than programming, that is the real power to change the world. If you want to get those broader concepts across, you could start at the section of our website on computational thinking. It's at [www.cs4fn.org/computationalthinking](http://www.cs4fn.org/computationalthinking). The teacher's area mentioned above is intended also to provide resources for teachers to get across broader concepts, especially the syllabus area. Other invaluable resources for teachers that have a similar philosophy to ours can be found at the CS Unplugged website ([csunplugged.org](http://csunplugged.org)) from the University of Canterbury in New Zealand, and the University of Glasgow's CS Inside project ([csi.dcs.gla.ac.uk](http://csi.dcs.gla.ac.uk)).

## YOUR FIRST ARTIFICIAL INTELLIGENCE PROGRAM

Programs are just instructions about actions to follow. They must also give the



order in which actions must be followed so there is no confusion about what is intended. To write a program, just click on the action you want to add to the program in turn, then click on "Add". You'll find this applet in the FUNdamentals section on the cs4fn website.

## THE FRENCH PEASANT'S LOCK AND GRAY CODE



The French Peasant's lock puzzle has appeared in lots of different forms: "The Chinese rings" puzzle being the most famous. There is even a version with elephants to get in a line and march off.

The solution to the lock is

actually something known to Computer Scientists as Gray Code: a code used in modern digital TV. Whatever their physical form, all the variations of the lock puzzle have the same solution and are logically (and so their solutions algorithmically) identical. Solve one and you've solved them all. Computer Scientists love pulling that trick with problems! You can download the puzzle and read more about Gray Code at <http://www.cs4fn.org/binary/lock/>

*I understand you also do outreach work in schools. How is that organised? What sort of things do you get involved in?*

A big part of cs4fn is live events. After all, we started at a careers fair. Now we visit schools to give talks about subjects ranging from women in computing to artificial intelligence. Most of our visits happen around London, but for larger audiences we can go anywhere in the UK. To organise a talk, teachers should visit [www.eecs.qmul.ac.uk/public-engagement/lectures-for-schools](http://www.eecs.qmul.ac.uk/public-engagement/lectures-for-schools), where they will find a list of our talks and contact details. Be prepared: you will need to book at least a month or two in advance. We're really busy!

***As cs4fn has grown, it's also led to 'spin off' developments. Could you outline what is now available? Who are these new resources trying to reach?***

We've produced spin-off magazines about related computing subjects. The first was Audio! which is about audio engineering and music technology. It follows the cs4fn pattern but draws on a mix of music, computing, electronic engineering, and physics. More recently we've produced ee4fn magazine that focuses on physical computing and electronic engineering. We

hope they also inspire students about computing topics, they are just coming at it from different angles. We've always aimed to build bridges and show links between subjects.

Peter has also produced some spinoff magic books. There's one where the tricks are all mathematical – the Manual of Mathematical Magic. You can find that at [www.mathematicalmagic.com](http://www.mathematicalmagic.com). He has also written one called Illusioneering, where the tricks centre on general science and engineering. That's available on the web too, at [www.illusioneering.org](http://www.illusioneering.org).

***Now is an exciting time to be a teacher with an interest in Computer Science. Do you have any future plans you could share with us?***

We're really excited at the moment that this spring and summer we will be offering some CPD courses to ICT and computing teachers. These will be geared for GCSE teachers looking to learn about programming and core concepts in computing. We will also talk about practical ways to teach these topics in class. We will post news about these courses at [www.cs4fn.org/teachers](http://www.cs4fn.org/teachers). We'll make sure to tell CAS members about them too!

*Peter, many readers will also know, from seeing you at CAS conferences, that you are a magician. Indeed cs4fn have*

*produced two volumes of tricks for teachers to use. What's the link between magic and computer science?*

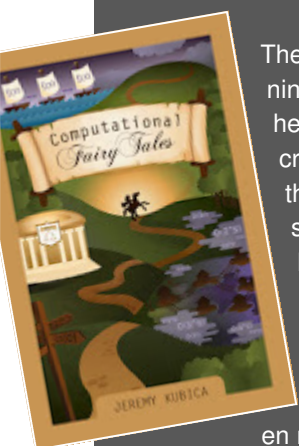
Magic tricks are just like computer programs. There's an algorithm – a list of instructions that contains the 'secret' of the trick. Follow them and the trick works just as if a computer follows a program it does the right thing. Then there's an interface, a layer of performance, between the instructions and the audience. Without a good performance a magic trick isn't satisfying and may even fall flat, and the same is true for programs. Without well-designed human computer interaction programs can be unusable. Without a good interface a program isn't worth its own code. Both magic tricks and programs need to be built on the same understanding of human psychology.

Like you said, we've written two books of computer science magic tricks. Both start by presenting a trick, then telling you how it's done and the computer science behind it. You can download the books in pdf and read more on the site at [www.cs4fn.org/magic](http://www.cs4fn.org/magic).



## INTRODUCTION TO COMPUTER SCIENCE THROUGH FAIRYTALES

I once read there are only 7 stories, and every book, film or play ever written is a variation of these themes. Most stories and films my children enjoyed have been a variation of the quest theme (think Jack and the Beanstalk, Shrek etc.). We all (I think) like a good story. Reading a story is a good deal more interesting than reading a text book, but it's entertainment rather than learning. Computational Fairytales, by Jeremy Kubica, is a rare book that manages to cross the divide. It is a classic quest. Ann, the daughter of King Frederick must rescue the kingdom from the coming darkness. She visits places and people in the kingdom such as the town of Bool and the city of G'raph, where travelling salesmen have enormous problems. Each chapter works as a standalone story and I have used some (such as Sailing and Functions) with my GCSE class. Online at <http://computationaltales.blogspot.co.uk/> the stories are freely available for classroom use under Creative Commons.



The book starts at the beginning of Ann's quest, and helpfully, the chapters increase in complexity throughout the story. For some younger pupils the later chapters may be a little complex - but try it - your junior school pupils may surprise you. From my point of view, it's given me confidence. The concepts are explained so clearly it gives me the basis of how to teach. It also spins out into ideas for reinforcing kinaesthetic activities. Try the chapter on Bullies, Bubble Sort, and Soccer Tickets then get your class to bubble sort themselves in height order. The topics go from the basics of IF to much more complex stuff such as Dijkstra's algorithm and covers topics on A-level syllabuses in chapters such as Binary Searching for Cinderella. The fact that the protagonist, Ann, is female is perfect. Nothing really is made of this in the story. We don't have any references to Ann having beaten all the boys in her class or anything nauseating like that. She's just a person who has a job to do. *Lucy Bunce*

## SEVERAL DIFFERENT METHODS FOR TEACHING PROGRAMMING

**Computer Science Education research is thriving but with such a large volume of research papers it's often hard to know where to begin. Peter Donaldson provides a short overview of recent developments in pedagogy.**

Around the world University based researchers are carrying out new studies and trialling teaching techniques designed to broaden participation in Computing and develop students' understanding. They provide many insights into more effective methods of teaching programming.

### Peer instruction with multiple choice questions

Doubling students' level of understanding compared with traditional lectures might sound unlikely but that's just what a number of different CS educators have achieved using Eric Mazur's method of Peer Instruction. Background reading and a pre-lecture quiz prepare students for a session that involves voting on a probing multiple choice question. They discuss their answers in small groups and a group decision is followed by another vote. Finally a whole class discussion about why one answer is correct and others are not leads to increased understanding for all.

### POGIL a process of orientated and guided inquiry learning

Discovery learning can sometimes be a frustrating experience for students who haven't built enough meaningful links between concepts. These are the students who benefit most from this approach. POGIL provides a team based, three phase structure. Specific roles for each student models the original process of discovery and research. Teams have questions that guide them in exploring existing data and models. They attempt to explain what they've found, define a concept and then apply it in other contexts.

### Abstraction transition taxonomy aids effective question design

Students often have to answer programming questions at three different levels of abstraction, English, CS speak and Code. Questions posed at one level that have answers at another can develop a deeper understanding of computational concepts.

### Subgoal labelled instruction to strengthen learners' mental models

Small changes to the way we do things can make a huge difference to effectiveness. When labels, such as creating a new component, appeared in ApplInventor video tutorials it not only improved the time to complete the task but gave students the ability to transfer what they'd learned to other tasks weeks later. The students even talked in terms of subgoals (I need to create a new variable for example).

### Pseudocode as a means to assess fundamental CS concepts

A number of different disciplines have assessments that reliably measure the level of understanding of their core concepts. With programming most teachers have had to rely on questions posed in a specific language but do they really test the fundamental ideas or just students' ability to remember a particular set of syntax? In developing an introductory course Dr Allison Elliot Tew proved that a carefully designed pseudocode could provide just as reliable a measure of the core concepts as questions designed using a specific programming language.



# PROBLEM SOLVING APPROACHES USING **ALGORITHMIC PRINCIPLES**

**João Ferreira lectures in Computer Science at Teesside University. His thesis “Principles and Applications of Algorithmic Problem Solving” offers educational material demonstrating how this may be used in the classroom. Lyndsay Hope takes a look.**

Beginning with the idea that effective teaching depends on an abundance of varied materials, Ferreira developed a range of Teaching Scenarios with a recreational slant that aim to stimulate self-discovery. Ferreira writes “Based on the material developed, we are convinced that goal-oriented, calculational algorithmic skills can be used to enrich and reinvigorate the teaching of mathematics and computing.” He supports his arguments with fully worked examples of problems.

The author discusses principles of, and techniques for, algorithmic problem solving. He explains how such principles may be used with students. The appendix contains several scenarios; these are clear, well structured and engaging.

Each teaching scenario involves fully worked out solutions to algorithmic problems with detailed guidelines on the principles encapsulated by the problem, guidelines to tackling it and an explanation of how it is solved.

Each begins with a description and summary goals. A Problem Statement provides a succinct outline. The Students Should Know section states requirements which help teachers evaluate the scenario’s appropriateness for their pupils. The Resolution section presents a fully worked solution together with explanatory notes.

‘Notes For The Teacher’ explore core components in detail; recommendations are made on presentation of materials with interesting discussion on introducing key concepts, and questions that the teacher should or should not ask. These follow Polya’s recommendations on unobtrusive questioning to promote self-discovery.

Exercises are given for homework’s, together with solutions. Extensions provide useful material for differentiation. Recommended reading for the teacher and students are included.

Some scenarios cover new ground, while others are reassuringly familiar. The Island of Knights and Knaves is a pleasing introduction to calculational logic and Boolean equality whilst Will This Algorithm Terminate? could give an engaging introduction to program termination. Ferreira trialled the use of these materials in practical sessions with 135 first year students studying a “Mathematics for Computer Scientists” unit. In a workshop for Portuguese secondary-school students he also used the “Goat, Cabbage and Wolf” materials. Seemingly student feedback was positive, suggesting students enjoyed the recreational nature of the problems and that they enjoyed the challenge.

He expresses the desire to use his thesis as a starting point for a broader educational programme and to do more extensive study of the suitability of the methods explored. I can certainly see this material being extremely useful in introducing this subject to senior students. Resources are well structured, explanations thorough and lucid, further reading is relevant, and core problems are presented in an accessible manner.

I look forward to seeing where Ferreira will take this work – he is able to express knotty material in an engaging way and has a good understanding of what teachers need to open this area up to their own students. It is well worth a read so do go to the site for links to resources mentioned. <http://joaoff.com/publications/2010/thesis/>

## MY FAVOURITE **BOOK**

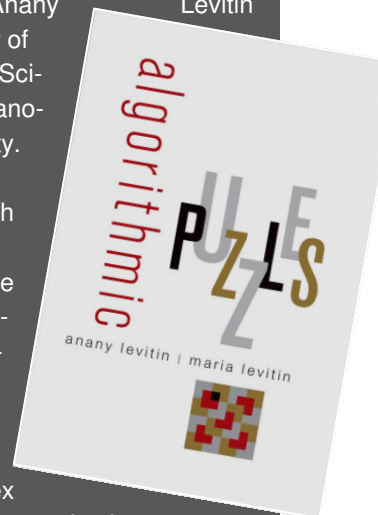
For those interested in developing computational thinking, this book is a beauty. It is, quite simply, a collection of 150 algorithmic puzzles i.e. puzzles that involve clearly defined procedures for solving problems. Anany

Levitin

is professor of Computing Science at Villanova University. The graded puzzles, with hints and answers, are supplemented by tutorials on algorithmic analysis and an index

grouping the puzzles by strategy. The authors’ stated aim is to ‘promote development of high-level algorithmic thinking (with no computer programming)...’ They succeed admirably. An excellent lesson resource—and a must for every Computing teacher’s bookshelf.

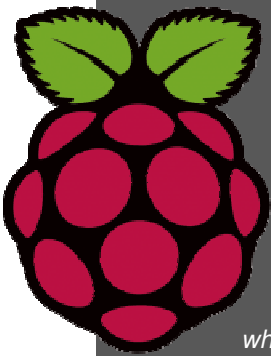
Roger Davies



## ITICSE CONFERENCE

The 18th Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE (pronounced it-eh-see) will be held in Canterbury at the University of Kent on July 1st to 3rd. There is a special teacher oriented day on the 1st July, featuring contributions from several CAS members: a keynote by Simon Peyton Jones, a panel session featuring Simon Humphreys and a workshop by Michael Kölling. There is a special day rate of US\$100 for UK teachers wishing to attend on the 1st, which should prove an interesting opportunity to gain knowledge and engage with researchers. More information can be found on the conference website: <http://www.cs.kent.ac.uk/events/iticse2013/>

## GOOGLE TO GIVE RASPBERRY PI TO ASPIRING YOUNG STERS



In a major development at the end of January, Google announced it would fund 15,000 Raspberry Pi to give to children. The announcement read:

*"The success of the BBC Micro in the 1980s shows what's possible. There's no reason why Raspberry Pi shouldn't have the same impact, with the right support." That was Eric Schmidt speaking in May about the opportunity for Raspberry Pi to inspire budding computer scientists.*

*Today Google is backing the Raspberry Pi Foundation with more than words, by providing funding to allow 15,000 UK kids enthusiastic about computer science to get a Raspberry Pi for free.*

*To ensure no Pi is wasted, devices will be doled out with the help of six educational partners: CodeClub, Computing At School, Generating Genius, Coderdojo, Teach First and OCR. Each organisation will have a supply of free Pi's to give to children they meet who demonstrate an aptitude and passion for computing. As an added bonus, each device handed out will come with a teaching and learning pack, created by OCR, and designed to help kids dig in right away and get the most out of their Pi's.*

*There's no magic solution to the UK's computer science education woes, but real progress is being made thanks to the combined efforts of many. Google is proud to lend our support and, we hope, a little Pi will go a long way.'*

Theo Bertram, Public Policy Manager, UK

As we went to press the details of how the scheme will distribute the devices to the six organisations mentioned (including CAS) hadn't been released. We all wait with anticipation for further details about this very welcome initiative. As soon as we know what numbers we will receive, details will be posted on the CAS Community Forum. Links to related blogspots from participating organisations can be found in the supplement.

# A MAGAZINE THAT SUPPORTS THE NEW BEDROOM CODERS

**The MagPi is produced by Raspberry Pi users, for Raspberry Pi users. The first issue appeared in May 2012, at a time when most of their writers had not yet received their slice of Pi. Mark Tranter was one of them.**

It's a measure of the Raspberry Pi success that, less than a year on, there are over a million boards in the hands of users. Our first issue kicked off with a history of Raspberry Pi development, but quickly moved on to more practical matters - how to set up an emulated Pi, the robotics project "Skutter" and, of course, programming in Python and Scratch. As the months went by, The MagPi continued to offer an ever-more impressive range of material. The hardware articles have always been a strong point: the hugely popular "Skutter" robot featured in several issues, as did articles on using GPIO pins for a range of projects including a "Santa Trap", a temperature sensor and interfacing with an Arduino. Readers were given in-depth analysis of the components on the board as well as instructions on how to set up an SD-card and get the necessary peripherals connected. Programming maintains a high profile: The MagPi has featured articles (always including sample code) on Ada, C, C++, CESIL and SQL. Python and Scratch have not been neglected, of course! Many Raspberry Pi owners have little or no prior experience of Linux. Articles in the "Command Line Clinic" series, together with several pieces dealing with aspects of Debian, have helped users to become more comfortable with an unfamiliar operating system.

The MagPi has grown rapidly since the early days. As before, it is free to read online or download as a pdf. But Editor-in-chief Ash Stone and co. have worked hard to make printed versions available to buy. No Computing department should be without a set! <http://www.themagpi.com/>

Eric Raymond famously used the metaphor of the cathedral and the bazaar to characterise the contrast between the Open Source movement and its commercial counterpart. Cathedrals take a long time to build. They are expensive - symbols of power and orthodoxy. A bazaar, on the other hand, can spring up overnight. It offers people with little or no power the chance to engage in exchange with others. No single vision of the right way to do things can hold sway in the exciting chaos of a street market. Raymond's juxtaposition applies to education as well. Exam syllabi and school curricula are the building blocks of this "cathedral". But, in and around the official temples of learning, our students are inquisitively sampling from an intoxicating jumble of stalls offering entertainment, information and skills. Strutting about this bazaar, you might catch sight of a striking-looking bird. It is a MagPi, always searching for shiny

things with which to decorate its nest. Whether or not your students have yet bought a Pi, this magazine is sure to enthuse them and encourage them to get into some real computing. The Raspberry Pi has the potential to galvanise a revival of Computer Science in the UK. CAS members will be encouraging this renaissance in the classroom, but the real leap forward requires kids with their own machines, in their own homes and nothing to hold them back but their imaginations.



# FOSTERING CREATIVITY THROUGH CONTROL AND BASIC ROBOTICS

**Christine Swan, a CAS Master Teacher at The Stourport High School and VI<sup>th</sup> Form Centre urges teachers to provide opportunities for pupils to tinker and develop projects involving external control. The Raspberry Pi is one way into this world.**

I've always enjoyed finding out how gadgets work and building contraptions. I'm delighted the resurgence of Computing heralds a return to the days of innovative thinking and creative problem-solving. One key skill to be fostered in our students is the belief that anything is possible. Students develop resilience when experiments do not go according to plan. They gain experience in testing, troubleshooting and fault rectification. They also develop the ability to use simple starting materials to create something clever.

Lego sales recently recorded an increase of 25% partly due to sales of a line aimed at girls. I'm not sure that I'm happy about their "pinkness" but I am happy that girls are constructing things. Sales of the Raspberry Pi may well soon surpass the estimated 1.5 million units of the BBC Microcomputer. I have had people ask me: "What can we do with a Raspberry Pi?" to which I think it better to answer: "What CAN'T we do with it!"

The Raspberry Pi provides a wealth of opportunities for experimenting, tinkering, trying out ideas and pushing the boundaries of what is possible. I was tasked with thinking of activities suitable for use with GCSE classes. It wasn't difficult: chunks of the syllabus fit-

ted well and allowed demonstration of the Pi's capabilities. Computer architecture, networking, operating systems, programming in Python and investigating simple control systems are all topics that can be taught using a Raspberry Pi.

My final task was to create a robotic rover. I used a PiFace Digital interface board (developed by Dr Andrew Robinson at the University of Manchester School of Computing) to interface with the Raspberry Pi and older style Lego motors and wheels to create the running gear. The rover was easily programmed using Scratch and my students could quickly create programs to flash LEDs, sound buzzers and power the wheels. The PiFace uses simple screw terminal connections for input and output links, removing the need to solder and making it very quick to build new systems. Its simplicity and ease of use meant that students thought about what was required and were able to program without having to learn new skills.

At Advanced Level, the Raspberry Pi is an additional platform for students to experience. Its small size is similar to a mobile device and the System on Chip component provides an interesting topic for investigation. Students may have limited experience of the Linux operating system so the Raspberry Pi can widen their experience. They can create users, assign appropriate access permissions and investigate the outcomes of these. Even if they damage their operating system installation, it can be reinstalled or another SD card used. Networking the Raspberry Pi is straightforward and the terminal can be used to enter commands,



which is nearly always blocked for students in schools. The learning opportunities are rich and varied.

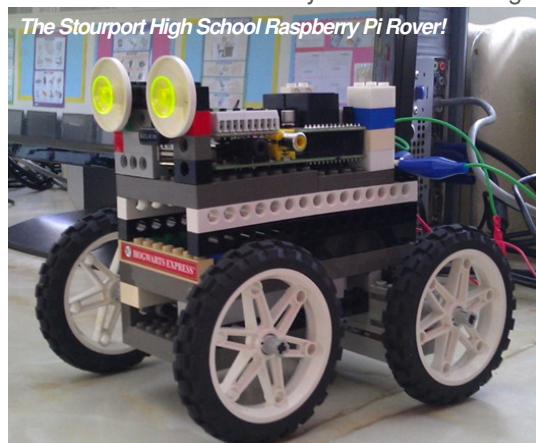
As we currently have limited numbers of Pis, when I use them in class, students have to work as a team and decide collaboratively what they are going to do. My role is to provide the spark of ideas but then, I'm a bit more hands-off. We need to cram a lot of learning into this year but of course, having fun is a great motivator. It is an exciting prospect that awarding bodies are also looking to embed the use of Raspberry Pi in their syllabi as an optional resource.

Take a look at the box below for suggestions of places to find further inspiration, using all sorts of resources, not just the Pi. I appreciate I am not training an army of robotics engineers, but my students are becoming adept at solving problems, thinking creatively, experiencing a new platform, working in a team and not giving up even when the going gets tough. In fact, the most difficult thing to find is time!

## FINDING INSPIRATION

For a quick starting project, take a look at Rob Bishop's Raspberry Pi Recipes at <https://github.com/RobBishop/RaspberryPiRecipes>.

I love Carrie-Ann Philbin's Geek Gurl Diaries website which is at <http://www.geekgurldiaries.co.uk> and the endless stream of fantastic ideas on Lady Ada's site: <http://learn.adafruit.com/> There's also a growing number of resources on the CAS community.

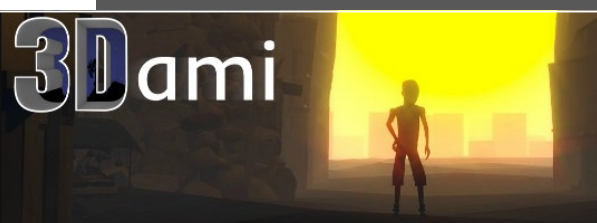




## 3-D ANIMATION COMBINES COMPUTER SCIENCE AND ART

There is much talk of STEM, but there is a growing recognition of STEAM: Science Technology, Engineering, **Art** and Maths. The United Kingdom has one of the world's largest digital arts and games industries, Tomb Raider and even Batman (to a large extent) are British. Yet despite this there is little formal education available for pre-university students, and limited awareness outside the industry of how to break in. 3-D animation offers a perfect combination of Computer Science and Art. The ability to break down a large scene into its components, the use of algorithms to generate cityscapes, is all computational thinking. After the Next Gen report I was fired up to try to help my aspiring Sixth Form 3-D animators realise their dreams.

The first step was to set up a Wednesday afternoon animation club using Blender. Within a few weeks I was completely out-classed by my students. Four entered Animation12, coming runners up in their category. It was time to get professional help!



Last summer 3Dami was founded. Borrowing ideas from Young Rewired State, eleven 16-18 year olds set up a small-scale animation studio in some empty London office space. Their task? To develop a 3-D animated film in 7 days. Over the duration professional mentors, academic experts and industry visits gave students the skills and knowledge to break into the industry.

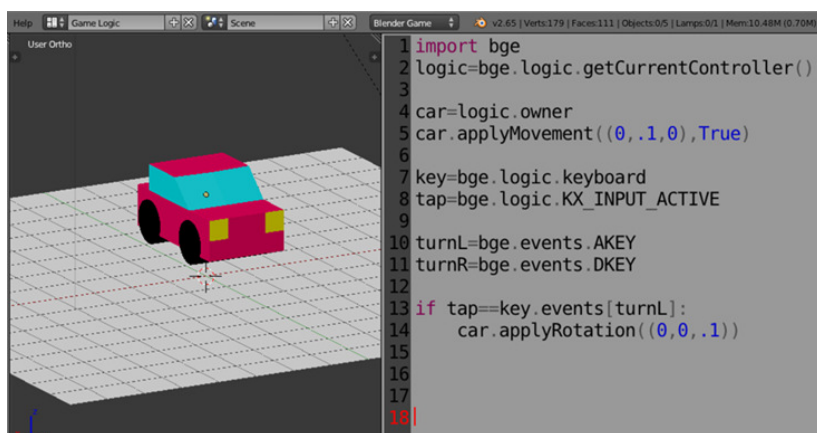
This intensive experience was supported by NESTA, Teach First, Escape Studios and Double Negative (the guys behind the Dark Knight!). A combination of artistic skill, teamwork and some haribo fuelled late nights in the office produced "We need more time". You can check out their work on the website. We're recruiting for 2013 so if you have any students aged 15-18 who you think would want to get involved get them a copy of Blender, and apply at 3dami.org.

*Peter Kemp*

## PROGRAMMING IN PYTHON USING BLENDER'S GAMEENGINE

**Computing should be fun, right? So why introduce a programming language using a text based editor and string manipulation exercises? Ben Smith, Head of Computing at AKS in Lytham St Annes suggests another approach.**

We all know kids today take 3-D games with a gazillion colours for granted. Blender is a free, open source, 3-D programming environment that benefits from an immediate Wow! Factor. A typical response within the first few minutes is an audible "whoa!" as they orbit and zoom in on the default cube. Blender uses Python. With a click of the mouse it can place highly detailed 3-D graphics right next to a python editor. With four lines of code you can have your 3-D rocket ship blasting off!



The interface is much like Flash, so a little scary at first. But once accustomed you soon realise the myriad extra options filling Blender's recesses are natural areas for gifted and talented to play. The built in physics types keep them interested for hours once they realise they can make their objects soft like jelly or float like balloons. However, my main aim is always to bring things back to the Python editor. As every couple of lines of code improves their creation's functionality they really want to learn. Floats, Conditional Statements and Loops are easily accomplished. The module I teach reinforces these basic concepts with a succession of simple games building on the lessons from the last. To teach string manipulation why not have the words come from the mouths of a game character? What better motivation for incrementing an integer than have it effect the speed of their own virtual sports car. Need to teach for loops? Throw a banana skin on the track and use it in a FOR loop to monitor how many spins the car does before it comes to a stop.

It's three years since a pupil introduced me to Blender. A simple game can be made in minutes. The ease of seeing impressive results is why I think it's a winner. Imported models can be linked to code to take control of anything from a racing car to a cheeky blue hedgehog. Small 10-20 line snippets can focus on particular programming skills. Want an inventory – here's how lists work. Want to add a second level – global variables will allow that. I have a series of introductory videos (<http://goo.gl/QWYWS>) aimed at KS3 with lesson plans. Thousands more YouTube videos can fire up their imaginations. For example, why not wrap a photo of their face around their main character to become part of their game? Pupils will be racing back at breaks to continue working. Mine often arrive with a new project they can't wait to show me and their friends.

# RETAKING CONTROL: KEEPING TRACK OF CODE MODIFICATIONS

**Since programming started the management of the stuff of programs (the source code) has been the stuff of nightmares. Version control systems were developed to help with these issues. John Stout, who teaches at KGV College, Southport explains.**

Student: "My program's stopped working!". Teacher: "What did you change?". Student: "I can't remember. It was something to do with the file-handling!" Aaaargh! We're all familiar with the problem. If you are a member of a large programming team then the issues get even more complex. If you develop different program versions (different operating systems perhaps, or with different features) even more problems can arise. In schools the first scenario is common, particularly in coursework units.

So how does version control work? When a version control system is installed it creates a *repository* to hold copies of any programs under version control. When the first working version is created the *local copy* (perhaps in the programmer's local drive or network area) is *committed* to the repository with comments, e.g., *first working version*. From that point, when a file is committed, the changes required to make the new version reproduce the older version are recorded (known as *reverse-deltas*).

You can now *compare* your *local copy* (known as the *head*) to find out what changes the student made to their file-handling code as in the example above. In some systems the head is stored completely for efficiency reasons. If you need to *revert* to a previous version ("Well, I thought that re-writing the file-handling module was a good idea, but the new version is even worse") then the reverse-deltas (between the head and the version you want) are applied, starting with the head, and the result overwrites your local copy. One student described GitHub, a web-based system, as "Undo for programs", which sums it up rather neatly.

The commands (*commit*, *compare*, *revert* etc.) can be carried out from the command line but most development environments have plug-ins that simplify the process and let the programmer carry out version control from within the environment.

When a programmer working in a team project wants to make modifications to a project file they have to *check-out* the file from the repository. Nobody else can make changes to this file (like database record locking) until a *check-in* of the file. More usually multiple programmers can check-out the same file but on check-in a *merge* of their changes and any others already checked-in is made (often automatically).

There are many version control systems available. Subversion is a popular open source system. An open source system, it has good integration with several interfaces: TortoiseSVN for Windows Explorer, VisualSVN and AnkhSVN for Visual Studio, and Subclipse for Eclipse users. It's also available for Mac and Linux.

Students can adopt simpler approaches. For example, every time they save a file they should backup the previous version. Some software, e.g., Notepad++ will do this automatically, appending the date and time. This can be a bit of a nightmare though, with multiple files in a project, when it comes to restoring an older version.

The Eclipse IDE has a *local history* facility that does this and there's at least one similar add-on for Visual Studio. These will also integrate with *compare* tools that let you compare versions. For single student developers this is probably enough.

## YOUNG REWIRED STATE

Young Rewired State is a network of software developers and designers aged 18 and under. It is the philanthropic arm of Rewired State and its primary focus is to find and foster the young children and teenagers who are driven to teaching themselves how to code, how to program the world around them. The aim is to create a worldwide, independent, mentored network of young programmers supported through peer-to-peer learning and ultimately solving real-world challenges.

The Festival of Code is our annual celebration of everything code. It occurs all over the UK every year in the first full week of August, and ends with a long weekend at the Custard Factory in Birmingham (UK), with everybody coming together for a weekend of showcasing the amazing achievements created during the week. Expect talks from awesome and influential people, pizza, as well as games and table tennis.

This year the Festival of Code will run from August 5<sup>th</sup> to 11<sup>th</sup>, all over the UK. It is a wonderful week long event in the Summer holidays that gives new students a great introduction to the community and enables them to decide whether they would like to be a part of further events and activities.

This is the fifth annual Festival of Code and it continues to grow in popularity. YRS gives an outlet to developing talent; to further their understanding and exposure to open data and provide a creative and fun atmosphere for them to meet like-minded individuals. Make sure your budding coders are aware of it and sign up: <https://youngrewiredstate.org/>



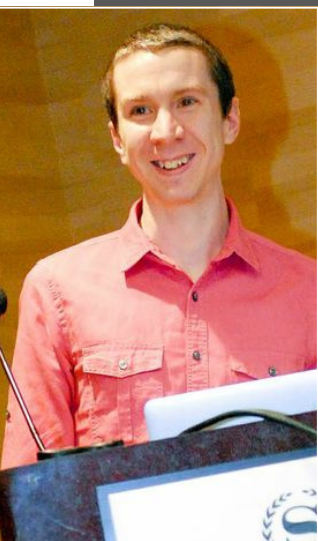
Michael Kölling

# OUTSTANDING CONTRIBUTION TO COMPUTER SCIENCE EDUCATION

**CAS founder member, Professor Michael Kölling of the University of Kent was given the annual award for Outstanding Contribution to Computer Science Education at the recent SIGCSE conference in the USA.**

## SHARING THE LESSONS OF CAS ACROSS THE ATLANTIC

SIGCSE (pronounced "sig-see") is a large computer science education conference held annually in the US by the ACM's Special Interest Group in Computer Science Education. The programme is a mix of research talks, panel sessions, workshops, and vendor sessions -- with ten sessions running in parallel throughout the three days! The usual attendance of around 1200 people is primarily university academics and teachers, and this year it was held in Denver, Colorado. CAS member Michael Kölling gave Friday's keynote speech on programming education tools (see right). Michael will be giving the keynote at this year's CAS conference, too, so you'll have chance to hear him speak then.



I presented a paper all about CAS. The US faces similar challenges: promoting CS within the curriculum, training up sufficient teachers and supporting existing teachers. My hope was to showcase work CAS members have done in areas such as the hubs, CPD, CAS Online, and our successful lobbying. A rehearsal video is online [here](#).

Some of the best sessions included school students as speakers. In one, we heard from two ten year-olds who had been using Scratch. In the words of one: "I decided that using a poster or PowerPoint for my book report was too boring, so I made a series of mini-games to illustrate the key points". Poster and PowerPoint too boring... academics take heed! *Neil Brown*

Michael said: "It's a great honour and quite humbling, looking at the list of previous recipients" -- the international award has previously been awarded to computer scientists such as Edsger Dijkstra, Grace Hopper and Niklaus Wirth. Michael was present at the first CAS working group meeting -- and has been a frequent contributor to CAS ever since. He has given workshops at the CAS conference, run CAS training courses for teachers, and most recently dedicated time and resources to designing and developing the CAS Online Community which has become so popular with CAS members. Michael will again be giving a workshop at this year's CAS conference, and also giving one of the keynote talks.

Michael's journey from his native Germany to the UK took the scenic route, with a PhD from Australia and time spent working in Denmark before moving to the UK eight years ago. His work on beginners' programming environments began during his PhD with the development of Blue, a massive project that including a new programming environment, a new programming language and runtime. Realising that a new teaching language was unlikely to see mass adoption, Michael took the key ideas of the environment across to the then-new Java language, creating BlueJ. BlueJ is now used by several million users each year, particularly in first year undergraduate university courses. Its associated textbook has sold over 140,000 copies worldwide, and popularised the objects-first pedagogy where object-orientation was introduced at the start of a course rather than viewed as an advanced late addition. Michael's next major project combined ideas from BlueJ with approaches from graphical "microworlds" to create Greenfoot. Intended to appeal to younger programmers, it provides an easy way to produce visible, engaging results with text-based programming in Java. Greenfoot is approaching half a million users annually, and developments are afoot for a major extension.

A more recent thread to Michael's work has been his work on implementing resource-sharing and community sites for teachers, based on his belief that educational software alone is not enough: it requires an entire ecosystem around it. This led to the development of the Greenroom to support teachers using Greenfoot. The Greenroom was my first project after joining the BlueJ/Greenfoot team, which we have since extended to form CAS Online, the community CAS members now use.

Applying ideas from graphic design and human-computer interaction to create beginner-targeted programming tools to provide easier pathways into computing characterises Michael's work. One of the keys to success for his tools has been to retain a minimal interface, even in the face of endless small feature requests that were independently reasonable, but together would have ruined the approachability of the tools. In his words, "my job managing these projects soon became saying no to people." So don't be offended next time a suggested feature for CAS Online is rejected -- it's part of a recipe for success! *Neil Brown*



# FINDING THE **HIDDEN TREASURE** IN LIFE'S GREAT COMPUTER GAME

**Suzy Race would like to tell you a brief story that involves good-ies, baddies, puzzles, sword fights and collecting treasure along the way. What's more, it does have a happy ending .... and no one dies.**

I spent my evening last night watching my boyfriend play the well-known console game, *Uncharted 3* and wondered why after 2 hours he could still be shouting at the computer in frustration? I realised this was the human instinct to succeed when faced with barriers or confusions. Surprisingly I can relate the story of *Uncharted 3* to my own journey becoming a lover of all things techy.

The story begins at 13 in an unfamiliar jungle environment surrounded by dial-up internet connections linking me to a maze of information and games to distract me from the on-going battle with homework. At this stage my parents are the baddies, as you can imagine, for not understanding that reaching level 5 in International Go-Kart Championship is far more important than Algebra equations.

I continued to work my way through school reaching each level and gaining the points I needed to continue onto the next stage in the story (sound familiar to Drake and Sully?!). At 16, once I had completed my GCSEs I was faced with my first major puzzle... pick the right A-levels for University. Such a puzzle is still well renowned for being one of the toughest at such an age. At 16 you live to plan the next

weekend, not what you want to be doing after completing 3 years at University. The school I attended did not offer ICT as a full A-level and back then you could not study what we know as "computing" today. So for a self-confessed computer geek there was no other option but study Spanish, French, Geography and Design Technology and leave the world of computing behind. But as I boldly stepped into a new world of language games and geographical formations I never lost my interest and passion in solving computer related tasks.

The next stage of the puzzle was going to University and getting a degree. At this stage the idea of doing computing was a distant memory and certainly nothing I considered or discussed as an option. I continued to explore a new map at Southampton University doing a degree in Population Sciences. After two years of baddies and sword fights I took a year out and worked for Unilever as a Business Analyst. During this year of high adventure to America and beyond I found the hidden treasure I had been looking for the whole time. A key to open the door into the world of Computer Science. I took a detour and left the Southampton map and I joined Oxford Brookes University to start,



and complete, a degree in Computer Science. I have now reached a stage where I have collected further treasures along the way, including being the Young Representative for BCS Oxford, local STEM Ambassador and having a job as a Graduate Trainee for a software company; Relayware.

The moral of the story? I wish computing had been pushed at school. Various forms of computing or IT are the basis for most things that we touch, interact with and buy. How can this not be fundamental to the teaching of the next generation of users? Today you cannot escape a social interaction without mention of the internet, a smart phone or a technological invention. Therefore, we must act to teach youngsters how to develop, manage and safely use any technology. I have engaged in the process of helping this development by becoming a STEM Ambassador to teach youngsters what I love about the world I work in.

The end of the journey is not yet known. However I do know I won't be spending 2 hours getting frustrated at my life / computer game because I've made the most of every opportunity along the way.

## COULD YOU BECOME A STEM AMBASSADOR?

Seeing and hearing computer scientists and other industry specialists can inspire school students to take their studies further. If you work in computing and feel you could spare some time to visit students in school you can find further details of the Ambassador scheme from the STEMNET website: <http://www.stemnet.org.uk/> STEM Ambassadors volunteer as inspiring role models for young people. They can contribute both to regular lessons or participate in extra-curricular activities such as STEM Clubs, Careers Days and visits. Of course, not all computing professionals will feel comfortable working with children, but you can still help build Computing in schools. Computing++ is an initiative that buddies up teachers and industry professionals to help develop, in particular, their coding skills. Sign up at <http://www.computingplusplus.org>



## Computing++

## MOVING FORWARD ON BOTH POLICY AND PRACTICE IN WALES

As always, there is much to report from CAS Wales: alongside increasing activity on the ground at each of the six Wales Hubs, we have also seen significant developments on the policy front.

In November 2012, Leighton Andrews, the Welsh Government's Minister for Education and Skills initiated a wholesale review of the ICT curriculum. In January, he announced the ICT Steering Group to undertake this review and report back in late June. CAS is well represented on this group, with myself as co-chair, secondary teachers Lucy Bunce (SE Wales Hub Leader) and Gareth Edmondson (SW Wales Hub Leader), as well as Professor Faron Moller (Director of Technocamps, Swansea University). This review has wide scope for reform of the ICT curriculum. While too early to release concrete details, the co-chairs had an extremely positive meeting with the Minister in March and have been encouraged to be bold and aspirational with the final recommendations.

A key issue in Wales has been the provision of CPD for teachers to upskill to teach computer science, especially in light of the success of the Network of Excellence in England. There is three years of ring-fenced funding for CPD via the Welsh Government's National Digital Learning Council, but this is pending the recommendations of the ICT review in June; more details to follow shortly.

2013 is shaping up to be an exciting year, especially with the focus on "Digital Wales" -- as you can see from the main article, the activity and initiatives coalescing around the LIFE programme in the Swansea region are hugely positive, with an explicit focus on computer science and developing the next generation of digital creators in Wales. And a final plea to encourage your colleagues in Wales to engage and join CAS -- spread the word! *Tom Crick*

## CAS WALES CONFERENCE

The annual CAS Wales / Technocamps Conference will be held over the weekend 5-7<sup>th</sup> July. Details [tocaswales13.eventbrite.co.uk/](http://tocaswales13.eventbrite.co.uk/)

# LIFE: TEACHING STUDENTS TO PROGRAM ACROSS SWANSEA

**LIFE – Lifelong Intergenerational Furthering Education is a flagship project for Swansea, promoting "Learning without Boundaries". Programme Manager, Simon Pridham explains the excitement behind the initiative.**

The programme has grown out of Casllwchwr Primary School in Loughor, Wales' first one-to-one iPad school, where the Head Teacher held the vision that technology should be as accessible as paper and pens. Over the past year, Swansea Council has worked closely with them to develop the programme for delivery across Swansea County.

The aims of the LIFE Programme are to: integrate mobile digital technology with a sound pedagogical base to enhance the key basic skills needed to raise standards in education; identify future digital leaders in partnership with Primary and Secondary schools as well as FE Colleges and Swansea University, in full conjunction with Technocamps, in order to equip the city with a skilled digital workforce; use creative technology to produce unique digital content through the medium of Welsh in full conjunction with TVchannel S4C; provide a digital platform in delivering essential technological skills, fully accredited and delivered for Teaching Assistants; research locally the effect of mobile technology on teaching and learning with pupils and young adults across the autistic spectrum; test the validity of its digital programme in socially deprived school environments; and create and develop an inter-generational skill share programme encompassing 21st century skills with established life skills, in order to develop cohesion in our communities across the generations.



Technocamps is a key stakeholder and working with LIFE. In particular, they are working together in two deprived areas of Swansea in both Welsh and English clusters, delivering twelve week app development courses with identified Digital Leaders, impressing upon them the importance of creating their own software through open source environments, giving them opportunities to work collaboratively across their communities. The results have been praised both from a content creation perspective as well as community cohesion piece of work.

The Summer will see the formation of the LIFE Talent Academy in full partnership with Technocamps. Pupils who have shown a high level of skill and talent during the twelve week programming and app development courses will be selected to form the LIFE Talent Academy. These pupils from socially deprived and targeted areas of Swansea will get free provision from Technocamps one evening each week at Swansea University to rapidly improve their skill set, and then return to their clusters and disseminate and empower others with such skills.

The Summer will see the formation of the LIFE Talent Academy in full partnership with Technocamps. Pupils who have shown a high level of skill and talent during the twelve week programming and app development courses will be selected to form the LIFE Talent Academy. These pupils from socially deprived and targeted areas of Swansea will get free provision from Technocamps one evening each week at Swansea University to rapidly improve their skill set, and then return to their clusters and disseminate and empower others with such skills.

The excitement behind where this programme can end up is quite phenomenal and by visiting [www.life1881.co.uk](http://www.life1881.co.uk) you will be able to understand why this is. You can also follow the programme on Twitter @life1881 and the Talent Academy itself @LIFETALENT.

# NORTHERN IRELAND: WHERE WE ARE AND WHERE WE ARE GOING

**Clarke Rice provides a summary of his observations on the direction of travel for the ICT and Computing curriculum in Northern Ireland, along with valuable background information for those on this side of the water.**

Education is divided between two ministers in our Assembly. School (4-18) is looked after by the Department of Education NI (DENI) and Minister John O'Dowd. FHE, which includes 16+ FE Colleges are the remit of the Department of Employment and Learning (DEL) and Minister Stephen Farry. Both have said Computer Science is a good thing. In 2012, O'Dowd asked CCEA to examine the need for Computer Science throughout the curriculum. Meanwhile the complaint of Farry, FE and Universities is that students arrive to do Computer Science thinking it will be like ICT. FHE points the finger of blame at the school curriculum. CAS have talked with politicians requesting a clear action plan from DENI.

C2k, the managed network service, will soon become the *Educational Network for Northern Ireland* and the tools needed to support Computer Science to A-level are being considered. In discussion with CAS, we have requested a range of programming tools to be supported as standard.

In Primary schools a new, less-prescriptive curriculum now allows teachers more flexibility to innovate. A growing number are using tools such as Scratch and end of KS2 assessment, designed by CCEA, now includes some programming tasks.

There are major concerns at KS3, where assessment focuses on the use of cross-curricular ICT skills. So, while the scheme allows pupils to design a 'Chop the Head off King Charles' app, few history teachers have the skills to teach or assess this. Hence, the concerns that the scheme may aim to the lowest common denominator. Unions urge a boycott of this scheme.

At KS4 CCEA's GCSE ICT now allows programming. A games-development task can be interpreted to allow Scratch, Greenfoot, GameMaker etc. A growing handful of schools offer GCSE Computer Science (OCR or AQA) but CCEA currently have no plans to offer a GCSE themselves. Nonetheless the CCEA ICT course is sparking increased demand for programming.

CCEA withdrew A-level Computer Science in 2002. In most schools, CCEA ICT is offered in its place. However, CCEA have developed a new Software Systems Development A-level for delivery from September. This is a major, and welcome, change in direction. With more schools offering Computer Science at GCSE, A-level demand is likely to increase. Successful delivery though will depend on improved provision from C2k as many schools simply do not have time to work around the network restrictions currently in place.

STEM funding was recently made available via the Southern Education and Library Board, for a small number of ICT teachers. This included 3 days of training (including Python, C#) and 2 days of placement within a software development company. However welcome this is, more training, for more people, is clearly needed. Some teachers are investing their own time and money in training, as they are keen to teach Computer Science. However, it is not feasible (or fair) to expect all teachers to do this – especially those for whom Computer Science is entirely new. Advances in the perception and demand for Computer Science will be negated if the subject cannot be delivered effectively. These are the current challenges ahead.

## SWITCHEDON SCOTLAND ISSUE TWO PUBLISHED

The second issue of our CAS Newsletter specifically aimed at members in Scotland, was published in January. Double the size of the first issue, it leads with a report on the hugely successful CAS Scotland Conference. There was an abundance of excellent sessions and links to videos of many of them can be found inside.

These will be of interest to teachers outside Scotland as well, such as Muffy Calder speaking on 'Computational Thinking In the Digital Age' and Quintin Cutts discussing 'Teaching Programming: Too Much Doing, Not Enough Understanding'.

Further features on Tackling The Gender Gap, Adobe Generation, the wonderful RSE/BCS teaching materials developed by Jeremy Scott and exciting news from Run-Rev about LiveCode will have a similar wide appeal. Profiles of the 'team' behind CAS Scotland reveal the breadth of experience and help put faces to names that may be familiar to those who participate in the CAS forums. Perhaps most important is the exciting sense of change running through the issue. CAS Scotland has grown quickly, with a third of Computing teachers already members. Chair, Kate Farrell outlines some of the challenges ahead and developments taking place on the policy front. You can download a copy from the CAS Community [/resources/542](#). Special thanks must go to editor Mark Tennant for another informative issue.





## A PAUSE FOR THOUGHT

Babylonian civilisation (C2000-600BC) mathematics worked in base 60 and gave us 60<sup>th</sup>s for hours and minutes, and 360 degrees in a circle. It also left us the very first algorithms for factorization, finding square roots and performing long division. Meanwhile in Ancient Egypt (C3000-300 BC) ancient Egyptians performed multiplication using an algorithm that draws on the binary system. Then the Greeks, who bequeathed such a rich mathematical heritage from Thales, Pythagoras, Democritus, and Aristotle, also gave us Euclid (C325- 270 BC) whose work *The Elements* is the earliest known attempt to formalise the concept of algorithm. The Euclidean Algorithm finds the greatest common divisor of two numbers.

The Sieve of Eratosthenes, attributed to Eratosthenes of Cyrene (C276BC-95BC) is a simple algorithm for finding all prime numbers up to a specific integer. He was also first to calculate the earth's circumference and the tilt of its axis. Archimedes (C287- 212BC) developed the first iterative method for calculating the value of  $\pi$ .

Muhammad ibn Musa Al-Khwarizmi (C780-850) was a Persian mathematician, some of whose work was based on Greek mathematics and Indian numbers. He wrote *On The Calculation with Hindi Numerals* and, when translated into Latin, his name was translated as *Algoritmi*, giving us the word 'algorithm'.

Algorithms are not a 20<sup>th</sup> Century invention. These examples are just a few of the crumbs in the long trail leading to Alan Turing's formalisation of the concept in 1936 through Turing machines and Alonzo Church's through lambda calculus. *Lyndsay Hope*

## BIOLOGY LOVES TECHNOLOGY

The slimy, messy world of biology and the clean, logical world of computing can make the perfect combination. In fact, sometimes it turns out it's the computing that's slimy and the biology that's logical. Computer scientists and biologists are teaming up, and finding that they are good at helping one another.

Our friends at cs4fn have teamed up with *Centre of the Cell*, an educational and health charity, to publish a book of facts from the combined worlds of biology and computing, called *Biology Loves Technology*. Teachers and students can order free hard copies or download a pdf. Each of the twenty snippets in the book is fleshed out in a full article on the cs4fn website. Together they make a wonderful introduction to the world of computational biology. You'll find further details at <http://www.cs4fn.org/biologybook>



## EXPERIENCED IN TEACHING COMPUTER SCIENCE?

CAS is expanding the Network of Teaching Excellence in Computer Science and is looking for experienced teachers to join our growing team of Primary and Secondary CAS Master Teachers. Are you an experienced teacher with a background in CS or related STEM discipline? Do you have a passion about sharing best practice and can engage with your peers in a professional environment? Do you have the ability to deliver practical workshops for teachers with appropriate course material? If so, we would like to hear from you. Please register your interest by emailing [mark.dorling@computingatschool.org.uk](mailto:mark.dorling@computingatschool.org.uk) (NoE CPD Coordinator), or [download the information pack](#)



## COMPUTING AT SCHOOL

EDUCATE · ENGAGE · ENCOURAGE

Computing At School was born out of our excitement with the discipline, combined with a serious concern that students are being turned off computing by a combination of factors. **SWITCHED ON** is published each term. We welcome comments, suggestions and items for inclusion in future issues. Our goal is to put the fun back into computing at school. Will you help us? Send contributions to [newsletter@computingatschool.org.uk](mailto:newsletter@computingatschool.org.uk)

**Many thanks to the following for help and information in this issue:** Alastair Barker, Jonathan Black, Neil Brown, Paul Brownling, Lucy Bunce, Ray Chambers, Tom Crick, Rik Cross, Claire Davenport, Roger Davies, Laura Dixon, Peter Donaldson, Mark Dorling, Dan Fraser, Peter Kemp, Lyndsay Hope, Toby Howard, Simon Humphreys, Catriona Lambeth, Brian Lockwood, Nicky Madams, Greg Michaelson, Sue Neiland, Reena Pau, Simon Pridham, Suzy Race, Clarke Rice, Sue Sentance, Ben Smith, Neil Smith, John Stout, Chris Swan, Mark Tranter, Andrew Walmsley and Michael Walmsley.

[www.computingatschool.org.uk](http://www.computingatschool.org.uk)

Computing At School are supported and endorsed by:



The Chartered Institute for IT  
Enabling the information society

Microsoft®

Research

Google™

CPHC  
The Council of Professors and Heads of Computing