So what are the implications for the practitioner, who is likely to have had little exposure in their training as to how to teach computer science to children...?

Supporting the new technologies curriculum in your school

Foreword
Our nation’s first Digital Technologies Curriculum has finally arrived, and if you have been paying attention in educational circles for the last two years, the buzzwords ‘Computational Thinking’ and ‘Coding’ are all the rage right now. So what are the implications for the practitioner, who is likely to have had little exposure in their training as to how to teach computer science to children, and is therefore either likely to be frightened or excited by what lies ahead?

The dawn of a new curriculum approach to technology
Australia’s first Digital Technologies Curriculum was finally endorsed in September of 2015, after a seemingly infinite wait from its initial conception in 2010 (The Department of Education and Training, 2015). A belated victory, the curriculum was stuck in an “awaiting endorsement” mode for nearly two years as the Education Minister at the time, Christopher Pyne, ordered a comprehensive review of the Australian Curriculum. Many keen teachers with an interest in technology would have wondered if indeed it would face the axe or not.

Fast-forward to December 2015 and the political landscape has a slightly different tune, where Prime Minister Malcolm Turnbull announced a five year National Innovation and Science Agenda (NISA) for Australia. Touted as a panacea for solving Australia’s apparent ‘innovation’ crisis, a significant part of the program is aimed at “ensuring...
the next generation of students have the skills needed for the workforce of the future...critical to ensuring Australia’s future competitiveness on the international stage” (Innovation.gov.au, 2016). The package, which brokers $51 million to schools, promises to deliver online computing challenges for schools, summer camps for students, and resources for assisting schools and teachers to implement the Digital Technologies Curriculum (although it is not exactly specified how this will be achieved).

Political duress aside, the talk about getting young people interested in computer science can only be a good thing. Advances in technology have led changes to the way people communicate, create, collaborate, solve problems and consume content. The increasing prevalence of technology continues to disrupt the way in which we work, play and learn. The challenge for schools is to acknowledge these societal changes and embrace them in order to support innovation and inspire students to learn and, ultimately, achieve.

**Australian Curriculum: technologies unpacked**

Upon first glance it appears that the curriculum has been developed in response to a changing educational landscape. The practicality of the Technologies learning area engages students in critical and creative thinking. The approaches to problem solving, experimentation, prototyping and evaluation instil into students the value of planning and reiteration to bring ideas to fruition.

The Technologies Curriculum encompasses two interconnected areas: Design and Technologies, where students use critical thinking to create innovative solutions for authentic problems; and Digital Technologies, where students use computational thinking and information systems to implement digital solutions.

The knowledge, understanding and skills in each subject are presented through two related strands: knowledge and understanding; and processes and production skills.

The aims of the Digital Technologies syllabus are to ensure that students can:

- Create, manage and evaluate sustainable and innovative digital solutions
- Use computational thinking and the key concepts of abstraction to create digital solutions
- Use digital systems to automate and communicate the transformation of data
- Apply protocols and legal practises that support safe, ethical and respectful communications
- Apply systems thinking to information systems and predict the impact of these systems on individuals, societies, economies and environments.

What is most promising about how this curriculum is written is the way in which it has embraced technology, as a holistic approach to thinking and exercising creativity. The traditional teaching of ICT in schools has usually been focused around integrating tools to assist in other subject areas, which is the intention of the ICT as a General Capability in the Australian Curriculum. Instead, the Technologies Curriculum paves the way for teachers to work with children as young as Foundation on pattern recognition and classifying data in contexts
that they can understand, gradually building up to the development of students with a strong understanding of computer science by the time they reach Year 10.

The content structure of the Technologies Curriculum can be viewed at australiancurriculum.edu.au/technologies/rationale.

Demystifying ‘coding’
Noticeable in the Digital Technologies component of the new curriculum are the ideas of Computational Thinking and Coding, which are introduced to students in early primary school.

The idea of coding is not to simply have students churn out computer programs. Rather, it is about assisting them to identify and analyse problems and to develop innovative and creative solutions, which will ultimately help contribute to a global society improved by technology.

Computational, system and design thinking all require the ability to examine problems clearly and to break them down into manageable parts in order to systematically analyse a process to best solve them.

It encourages the design of several solutions that can be applied in broad contexts. This type of problem solving – or thinking – is highly valued in the outside world. The ability to analyse problems and come up with clever solutions is the kind of thinking that continues to push our world forward, yet oddly enough, we don’t teach it in a deliberate and defined way – until now!

Through the Code.org (2016) initiative, more and more advocates are championing the idea of coding in schools, from celebrities like Will.I.Am to the Silicon Valley elite. The worrying trend is that the number of computer science graduates is currently not meeting demand, let alone in the future, when the demand is expected to further increase as the world starts to crave employees who are fluent in using technology to design products and solutions. Mark Zuckerberg is quoted as saying, “Our policy (at Facebook) is literally to hire as many talented engineers as we can find...the whole limit in the system is that there aren’t enough people who are trained and have these skills today” (code.org, 2016).

Coding can have the stigma of an un-sexy operation which takes place in a dark room with nerds sipping on soft-drinks, huddled around glowing screens and punching in lots of ones and zeros. Once upon a time, one was required to have a tertiary degree to operate punch-card machinery and to develop lines of code for programs that ran on mainframe computers. Through the advancements of technology, and particularly in the way in which we can interact with it, anybody of any age can now code.

Put simply, coding is about writing and following instructions. When a set of instructions is written for the computer, it follows them. Any time you have explained to someone how to bake a cake, or typed an equation on a calculator, or organised a filing cabinet in alphabetical order, you have essentially been designing an algorithm to execute a desired action. Coding is teaching a computer how to run a sequence of events for the reason that a computer can execute steps a lot faster than a human can.

Technology is starting to automate a lot of tasks that can easily be replicated by traditional human-driven processes. For this reason, we have started to see a shift in our modernised and globalised world.

Take for example Japan’s Toyota production line, which, through the use of machines and robots, can assemble a car to specific client orders in 18 hours; or the ambitious Google Car project, which promises to safely transport passengers from A to B without requiring the commuter to lift a finger; or the use of computer-assisted self-checkouts at the supermarket.

The overly critical may say that technology is taking over our jobs, which to some extent is true. However, more accurately, it is disrupting jobs and changing the supply and demand for workers. Jobs for production factories will still exist, as will people who drive cars, as will people who work in supermarkets.

What will probably be true is that these jobs are far more likely to require the skill sets of engineers and coders who are fluent with technology and programming, to be able to deliver solutions. Those who can build robotic arms to weld alloy will be more sought after than those who can assemble nuts and bolts. Those who can write programs that analyse traffic patterns for automated cars will eventually be in more demand than taxi drivers or chauffeurs. Those who can design computer-assisted checkout systems will replace those who manually scan items for consumers.

It is for this reason that we all need to embrace the new Technologies Curriculum for the good of our kids, and the future of Australia as a technologically relevant country.

How to support computational thinking, coding and the new technologies curriculum

Organisations and resources
Code.org: Launched in 2013, Code.org is a non-profit organisation that is dedicated to expanding participation in computer science, particularly by increasing participation amongst women. (See code.org.)

Hour of Code: an initiative of Code.org, is an annual event that promotes coding in primary and secondary schools across the globe. The coding tutorials can be completed online and have modules suitable for all ages. (See studio.code.org)

Code Club Australia: a nationwide network of free volunteer-led after-school coding clubs for children aged 9–11. (See codeclubau.org.)

Code the Future: aims to forge crucial links between the technology industry and education. (See codefuture.org.)

Bebras Australia Computational Thinking Challenge: Bebras is an international initiative whose goal is to promote computational thinking for teachers and students in Years 3 to 12, and is aligned with the new Digital Technologies
Curriculum. (See bebras.edu.au.)

*Computer Science Unplugged:* is a collection of free learning activities that teach Computer Science without having to learn programming first. (See csunplugged.org.)

*Careers with Code:* is a publication by Refraction Media and Google which promotes computer science careers in design, education, science, health, arts, media, law and business. (See refractionmedia.com.au/careerswithcode/ or search for Careers with Code on Google Play or iTunes App Store.)

*CSER MOOC:* the Computer Science Education Research Group at the University of Adelaide have developed a number of open, online courses designed to assist teachers in addressing the new Digital Technologies learning area. (See csdigitaltech.appspot.com/course.)

*Scootle:* is a national digital learning repository that provides Australian teachers with access to more than 20,000 digital learning items, provided by a wide array of contributors and aligned to core areas of the Australian Curriculum. Simply browse by curriculum area and select Digital Technologies to find matched resources and lesson plans. (See scootle.edu.au.)

*ACCE:* The Australian Council for Computers in Education have compiled a list of resources to support the implementation of the curriculum which have been developed in each of Australia’s states and territories. (See acce.edu.au/digital-technologies-resources.)

*CS First:* is a free initiative by Google which is aimed at increasing student exposure to CS (Computer Science) education. Suitable for ages nine to 14, materials and resources are free and online. (See cs-first.com.)

*Applications and tools*

*Move The Turtle:* is perfect for children five and up. Available on the Apple App Store. (See movetheturtle.com.)

*Blockly Games:* are a series of educational games that teach programming, and are designed for beginners with no prior experience. (See blockly-games.appspot.com.)

*Scratch:* is a programming environment developed by MIT (Massachusetts Institute of Technology) that can be downloaded or used on the Web. It is free and both basic and sophisticated enough to suit a wide range of learners. (See scratch.mit.edu.) There is also an iPad app suitable for the early years (ScratchJr). There is a whole community of resources and it is easy to find good lesson plans for integrating Scratch into the classroom. (See scratch.mit.edu.)

*MIT App Inventor:* Also developed by MIT is a smart phone / tablet app creation tool that transforms complex coding language into visual, drag-and-drop building blocks. With a simple graphic interface, even an inexperienced novice can create a basic functioning app within an hour or less. Runs on Android operating system. (See appinventor.mit.edu.)

*XCode7:* An alternative for developing on Apple’s operating system using the Swift language, XCode7 will install and run your designed apps on Apple’s iOS devices. There are also extensive curriculum resources and a community you can tap into. (See developer.apple.com/swift.)

*Hardware*

*Bee-Bots:* are colourful, easy to use, friendly robots that lend themselves well to teaching sequencing, estimation and problem-solving. Great for juniors! (See bee-bot.us.)

*Sphero:* is a cute little robotic ball that can be controlled and programmed on any smart phone. The latest edition is based on the adorable droid from the rebooted Star Wars franchise, BB-8. (See sphero.com.)

*Arduino:* is an open-source electronics platform that is suited to make interactive projects. Chipboards are relatively inexpensive and can take many types of inputs and sensors, lights and motors, and other actuators. Perfect for the upper primary and high school, Arduino is a useful starting point for educators who want to bridge software and hardware. (See arduino.cc.)

*Littlebits:* is a platform of easy-to-use electronic building blocks for creating inventions large and small. A circuit can be started in a matter of minutes by snapping a power module with inputs and outputs that are held together with magnets. There is no soldering required so they are perfect for young students. Kits can be purchased with various modules and accessories that lead to many types of circuit combinations. (See littlebits.cc.)

*Raspberry Pi:* is a low-cost computer that measures up to the size of a credit card. It can be plugged into a monitor or TV and also takes keyboards and mice. It can do most basic things that one would expect from a desktop computer but can also be used with other modules, or operated using the Python computer language. The website also has useful guides for learning, teaching or making projects with students. (See raspberrypi.org.)

*MaKey MaKey:* Touted as ‘an invention kit for the 21st century’, MaKey MaKey can turn everyday objects into touchpads and combine them with the internet. Any material that can conduct even a small amount of electricity can work as an input, so you and your students will have your tropical-fruit-drum-machine up and running in no time. Loads of fun with extraordinary possibilities! (See makeymakey.com.)

*References*

AustralianCurriculum.edu.au, 2016, Technologies:


*Further reading*