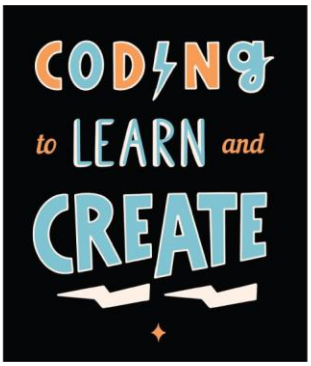

Dash and Dot

Introduction: Accessible Coding

Coding education is becoming an increasingly common and important part of the curriculum in elementary schools around the world. It is an activity that is recognized as supportive to the development of digital literacy, critical and creative thinking and collaborative skills, all of which are prerequisite skills for participation in the new knowledge economy. Though it is difficult to predict what the specific nature of employment positions will be in the future, the accelerated pace of change in today's market increasingly requires workers to be highly skilled in team collaboration, to be flexible and adaptive to new technologies and practices, to engage critically and creatively with digital innovations, and to embrace the demands of continuous learning and on-the-job skills development.

While coding is seen primarily as a means for developing the specific technical skills that will support future career opportunities, coding education is equally important as a way to develop social, daily living, communication, and creative skills—coding to learn. As Mitchel Resnick notes, "These skills are useful not just for computer scientists but for everyone, regardless of age, interests, or occupation" (Resnick, 2013). Students who are supported in developing self-reflection or metacognitive skills regarding their own learning needs are more likely to succeed in the learning process, and to ultimately be better prepared for navigating the constant change and lifelong learning required for participation in a variety of activities.

Coding education, when approached with this broad perspective and at an early age, offers a critical way for youth to develop the social, daily living, technical, and creative skills that will prepare them to contribute in the future.

Despite the recent emphasis placed by policymakers and educators on introducing students to coding at a young age, classrooms often do not have sufficient flexibility to accommodate the needs of students who have difficulty reading or who need additional guidance and time, or those who need help focusing or remembering. Many students who have complex learning needs are being excluded from the opportunity to learn or are relegated to passive roles, while their peers actively engage in solving problems together computationally. Most popular block programming environments are incompatible or untested with commonly used assistive technologies such as screen readers, magnifiers, and reading/writing aids, and are limited in their ability to be controlled efficiently using alternative input devices such as switches, eye gaze cameras, or onscreen keyboards. Often, programming environments cannot accommodate system-wide changes to magnification, spacing, or text size, and do not provide simplified

modes that help make it easier for students to read or focus on the salient parts of the program.

Yet students with disabilities often have the most to gain from the opportunities afforded by learning to code. Shared programming activities can help contribute to the development of skills such as expressive communication, literacy, sequencing, and metacognition—skills that are essential for learning as well as daily life and work. Perhaps most importantly, coding provides students with new ways to discover and create personal outlets for creative expression. Learning to code can empower students to be active producers of their digital environments, rather than just consumers of prefabricated apps and games. This empowerment is particularly valuable for those who depend on assistive technologies or personalized user interface adaptations. Programming literacy can provide them with new means to reconfigure, script, or develop their own personally tailored access tools and features.

While there are many educational coding tools available to support block programming for elementary school students, none of them were designed with the needs of students with disabilities in mind, and all have accessibility issues that limit their use by some students. The lack of user interface flexibility, interoperability with assistive technologies, and the inability to add customized learning scaffolds and adapted materials are all major accessibility barriers within these environments. Furthermore, the coding curriculum resources written for teachers are not geared to instruction of students who have complex learning needs, who benefit from highly scaffolded instruction.

As in other areas of the curriculum, taking a UDL approach to coding instruction and materials can provide greater opportunity for the active participation of students who have complex learning needs. Teachers need assistance in modifying coding learning resources to suit their students' needs and in understanding how to help students generalize coding skills to other aspects of curriculum and daily living.

This coding kit provides resources that take into consideration the need for scaffolded instruction and provides materials to allow for differentiation based on a range of physical, cognitive, literacy and visual perceptual abilities. This UDL approach considers multiple means of engagement, representation, and action and expression. To ensure the incorporation of multiple means of engagement, ideas and materials to assist teachers in planning and preparing students for coding are included. To ensure multiple means of representation, lesson design, strategies, and materials that meet the needs of a wide range of learners have been designed. Adapted materials ensure that all students have a means to demonstrate and express their learning in meaningful ways.

The kits have been developed with the use of robots to learn about coding and computational thinking, as opposed to self- contained digital programming applications because the use of physical, tangible programming has been shown to result in increased engagement and opportunities for meaningful collaboration (Lechelt, Susan, et al, 2018).

The kits are developed for popular robots being used in classrooms and include:

1. *Lesson plans that follow the scope and sequence outlined in educational resources provided by the manufacturer.*
   All lesson plans follow the same structure, including a description, objectives, vocabulary, and all of the whole class and individual materials required for the flow of the lesson (Anchor, Model, Practice, Apply, Share, Extend). The organization of activity under these headings allows teachers to tackle as much or as little of the lesson parts as is desired in a classroom schedule. Teachers need only prepare the materials for the part of the lesson they plan to complete on a particular day. The consistent lesson structure facilitates the development of a routine as teachers work their way through the lessons.

   While the lessons follow a scope and sequence that seems to be agreed upon by many in the coding field, teachers are encouraged to consider their learners and the difficulty of lessons, specifically between Lessons 9 and 14. Conditionals may be a particularly difficult concept to understand, so they may choose to cover the functions lesson prior to conditionals.

2. *Whole class instruction digital materials called out in the lesson.*
   These have been designed using Smart Notebook, PowerPoint, and or Google Slides.

3. *Whole class instruction print materials called out in the lessons.*
   These can be used to supplement the digital materials when technology is not available or for centre based activity.

4. *Individual print materials called out in the lessons.*
   Coding activity is usually handled through provision of a robot to a small group of students. The individual print materials are designed to provide all students in the group with an opportunity for active engagement during practice and apply activities. This helps to keep students engaged rather than experiencing wait times for their turn with the coding materials. It also provides teachers with a window into the thinking process for individual students who are working together in a small group to solve a problem.

5. *Individual digital materials.*
   Practice and Apply activities are provided in a variety of popular applications that incorporate alternate access (e.g. Boardmaker and Clicker). These are designed to support students who are physically challenged in manipulating print materials.

   While planning for coding lessons, you will find sample communication displays with core and fringe vocabulary to support students who do not have their own comprehensive communication system. The core vocabulary for these displays is taken, with permission, from the Core Project at the Centre for Literacy and Disability Studies. You will also see a

PowerPoint social script template to fill in with your specific location and staff – this type of resource, when used prior to starting coding activity, has proven helpful in allaying the anxiety that some students experience with new activities.

In addition to supporting students, teachers who are at different stages of knowledge, either in working with students who have complex learning needs or who are new to coding are supported. All lesson plans are scripted – teachers who have experience can use this as a reference while those who are new to coding may find comfort in having a script to follow.

All lessons follow the same format, so the routine becomes predictable for both teachers and students. Please refer to the Overview for a detailed explanation of each lesson component.

Tip sheets provide ideas to help students with communication in small group problem solving activity and sharing. Teachers will find multiple versions of each tip sheet. Choose the level of symbol support that your students require.

Students who have complex learning needs benefit from the explicit instruction provided in the lessons, as they learn to generalize these skills.

References:

Resnick, Mitchel. "Learn to code, code to learn." EdSurge, May (2013). https://web.media.mit.edu/~mres/papers/L2CC2L-handout.pdf

Lechelt, Zuzanna, et al. "Inclusive Computing in Special Needs Classrooms." *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18*, 26 Apr. 2018, doi:10.1145/3173574.3174091.