USING CODING APPS TO SUPPORT LITERACY INSTRUCTION AND DEVELOP CODING LITERACY

Amy Hutchison ■ Larysa Nadolny ■ Anne Estapa

Educators strive to ensure students are prepared for the 21st century. Learn how teachers can use coding apps to blend skill development and content learning to provide rigorous learning opportunities.

Over the past decade, literacy scholars have increasingly undertaken efforts to study the new skills, strategies, dispositions, intents, and social practices associated with digital technologies (e.g., Coiro, Knobel, Lankshear, & Leu, 2008; Lankshear & Knobel, 2007; Warschauer, 2006). A critical concern of digital technology as it relates to literacy is the idea that literacy, and what it means to be literate, is ever-changing because of the pace and constancy with which digital technologies emerge. Accordingly, these technologies change the ways in which we interact with text in its many forms (Leu, Kinzer, Corio, Castek, & Henry, 2013). In that vein, Knobel and Lankshear (2014) argue, "those who investigate new literacies try to anticipate beyond the present and envisage how best to educate *now* in order to enhance learners' capacities for effective meaning-making and communication in the foreseeable future" (p. 97). It is with this tenet in mind that we present and discuss coding literacy as an increasingly important and evolving form of literacy and examine mobile applications (apps) that connect to both literacy instruction and fundamentals of coding. Further, we describe how coding apps can be used to support literacy instruction and to develop digital and disciplinary literacy skills in the classroom.

Amy Hutchison is an associate professor of literacy education at Iowa State University, Ames, USA; e-mail amyhutch@iastate.edu.

Larysa Nadolny is an assistant professor of technology education at Iowa State University, Ames, USA; e-mail lnadolny@iastate.edu.

Anne Estapa is an assistant professor of mathematics education at Iowa State University, Ames, USA; e-mail aestapa@iastate.edu.

The Reading Teacher Vol. 0 Issue 0 pp. 1–11 DOI:10.1002/trtr.1440

The term *coding apps* refers to a category of apps that are designed to teach students the types of logical thinking, problem solving, sequencing, and planning required for coding computer programs. With many of these apps, students do not actually write code using programming languages; rather, they learn the concepts and types of thinking related to this practice. We provide examples of coding apps and the computational thinking they involve in the next sections.

Coding Literacy

In the field of computer science, programmers write code to give a computer step-by-step instructions on how to complete a task. This process is frequently called *coding*. The term *coding* originally applied to the act of creating in complex programming languages, but it is now also used to describe the creation of a sequence of instructions with tools basic enough for young children. For example, the programming language Logo and the well-known Logo turtle were created in the 1960s to expose children to coding concepts. As one of the pioneers of bringing computer science to youth, Seymour Papert (1993) envisioned Logo as a way to engage children in the creative process and the problem-solving process. But Papert (1993) wanted more than just computer languages and robotics in schools; he wanted students to "use programming as an expressive medium to study other topics rather than as a skill to be learned for the sake of learning it" (p. xviii).

Although experiences related to programming in K–12 schools have been around for the past 50 years, Papert's vision is now becoming a reality. Coding has moved beyond a skill for employment in high-demand fields and includes elements of expression, collaboration, and creativity. Proponents of this movement believe that developing coding literacy is a core competency for many types of 21st-century workers and that all students should have the opportunity to develop this type of literacy (Orsini, 2013). Maggie Johnson (2015), a director at Google and Code.org, urges us to consider the question, "Is coding now a fundamental literacy, like reading or writing, that all K–12 students need to learn as well?" (para. 3). We argue that coding literacy is an important type of digital and disciplinary literacy that is relevant to classroom instruction.

An important contributor to the increasing attention on coding literacy is Code.org, an organization that introduced the Hour of Code in 2013. During this introduction to computer science, anyone can participate in or organize a one-hour coding experience during Computer Science Education Week (e.g., December 7–13, 2015). During the Hour of Code, students participate in self-guided tutorials, targeted at students of all ages, that allow them to work at their own pace and skill level. Figure 1 illustrates one popular activity from the Hour of Code in 2014. Through a game-like format, students programmed the character Elsa from the movie *Frozen* to create a snowflake. Players drag and drop blocks of code to solve challenges and then advance through the levels of the game. A successful attempt will show how the code you created would look in a programming language, and failed attempts lead to tips and encouragement.

"We argue that coding literacy is an important type of digital and disciplinary literacy."

The Hour of Code initiative has reached a massive global audience since its inception in 2013 and has even received support from such prominent figures as President Obama (Mechaber, 2014). In 2014, 60 million students worldwide participated in the Hour of Code during the designated week (Code.org, 2014). Further, according to Code.org, through the Hour of Code, more girls tried computer science in 2013 than in the past 70 years. All of the tutorials, lesson plans, and recommended tools are provided on the organization's website (www.code.org). This initiative has grown to provide free computer science curricula designed for a variety of age groups and content areas.

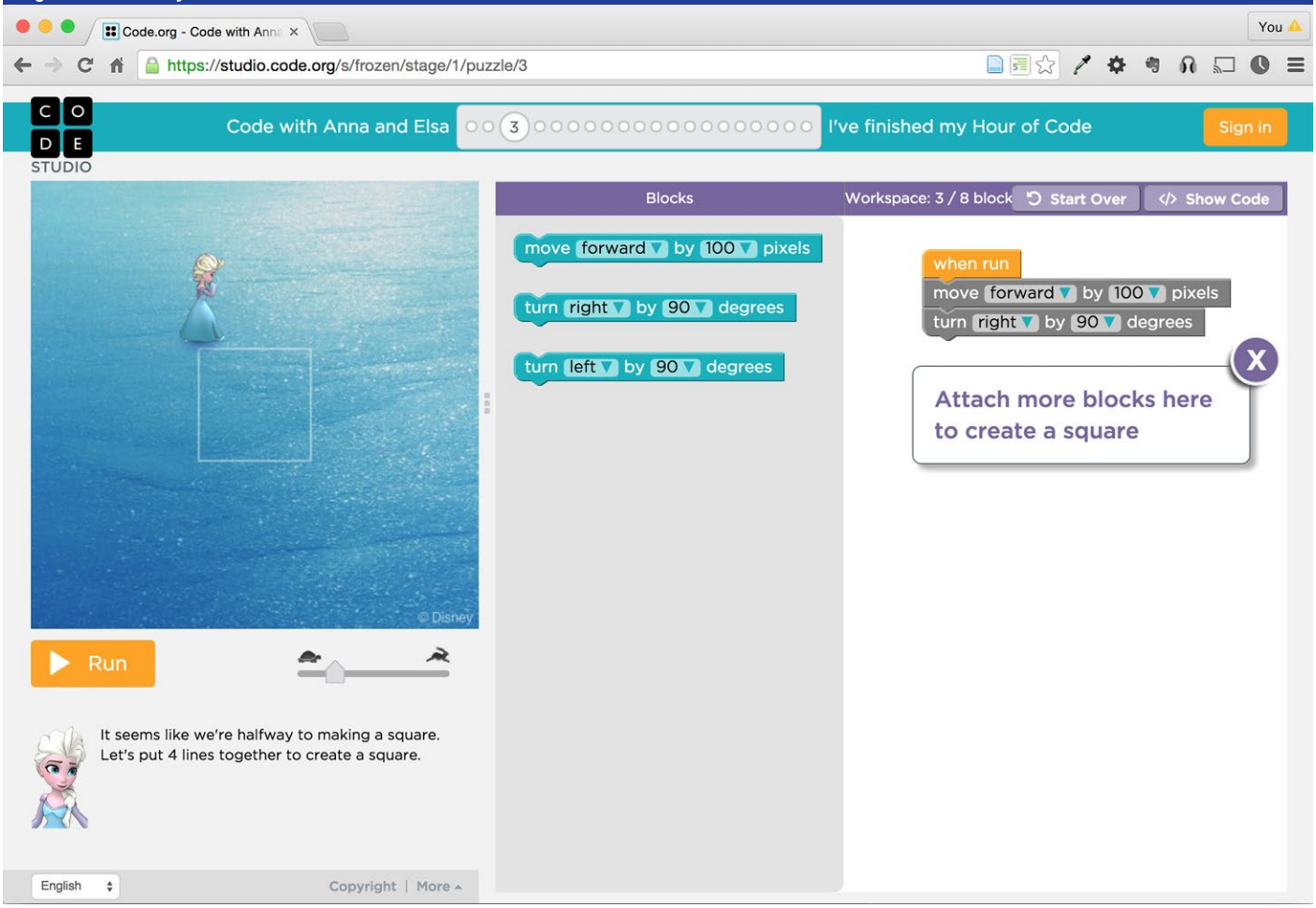
Using Digital Technology to Develop Specialized Language and Disciplinary Skills

Beyond the need to prepare students for the barely imaginable future, we believe, there are many tangible benefits of bringing coding apps into the classroom that support the development of literacy practices as they currently exist. First, many coding apps are designed as games, and as Gee (2013) has argued, well-designed games create problem-solving spaces with feedback and clear outcomes that lead to real, deep, and consequential

Pause and Ponder

- What type of support do you give students when they use iPad apps?
- How long do you play game-like apps before quitting?
- What kinds of skills and concepts can coding apps teach?
- Have you recently played around with technology to see what you can learn just by exploring?

Figure 1 Activity From the 2014 Hour of Code



learning. Gee (2013) contends that to participate in video games, players must learn the technical or specialized language of the game, which can prepare them to learn content-specific academic language in the future. Similarly, Knobel and Lankshear (2014) argue that many types of digital activities focusing on shared topics of interest often involve collaborative problem-solving communities and help students to "learn the 'ways of speaking' within a domain of activity and help them to participate more fully within it in terms of knowing what to ask for, contributing knowledge and knowhow, and becoming more 'expert'" (p. 100). This type of participation provides authentic and motivating contexts for developing specialized vocabulary that will be needed in the future.

In the same way, coding introduces students to the disciplinary literacies used by computer engineers, game designers, graphic designers, and more. Providing students with exposure to coding apps simultaneously provides them with the opportunity to learn specialized language and exposes them to the types of reading and writing performed in professions involving computer programming of any kind.

Coding to Learn

Further still, the development of coding literacy has even more immediate, and perhaps tangible, connections to traditional literacy. Instead of proposing that students learn to code, we propose that they can code to learn (ScratchJr.org, 2015). Initiatives such as the Hour of Code were originally introduced as a way to expose students to computer science and hopefully interest them in learning more on their own. However, we believe that, in addition to teaching the disciplinary literacies of computer science, coding apps and games also teach literacy skills. The developers of the app ScratchJr had this idea in mind. Accordingly, on their site they state the following (ScratchJr.org, 2015):

> Coding (or computer programming) is a new type of literacy. Just as writing

"Apps are most valuable for literacy learning when they are accompanied by well-designed lessons."

> helps you organize your thinking and express your ideas, the same is true for coding. In the past, coding was seen as too difficult for most people. But we think coding should be for everyone, just like writing.

As young children code with ScratchJr, they learn how to create and express themselves with the computer, not just to interact with it. In the process, children learn to solve problems and design projects, and they develop sequencing skills that are foundational for later academic success. They also use math and language in a meaningful and motivating context, supporting the development of early-childhood numeracy and literacy. With ScratchJr, children aren't just learning to code, they are coding to learn. (ScratchJr.org, 2015, paras. 4–5)

In this article, we expand this idea by proposing specific ways that apps designed to help students learn about coding can be used to teach targeted literacy goals while simultaneously addressing the evolving concept of coding literacy. As with any game, we believe, these apps are most valuable for literacy learning when they are accompanied by well-designed lessons and activities that are carefully planned to meet those goals. As Gee (2013) states,

> Kids learn with books or games (or television or computers or movies or pencils) when they are engaged in well-designed and good interactions with adults and more advanced peers, interactions that lead to problem solving, meta-critical reflection, and connections to the world and other texts and tools. (pp. 2–3)

Accordingly, in this article we describe several ways that coding apps could be used to design activities to support students' literacy development while building the skills and experiences of 21st-century learners.

Ideas for Integrating Coding Apps Into Literacy Instruction

Coding apps can be integrated into the classroom in a variety of ways and for a variety of purposes. In this section, we provide ideas for skills that can be built through coding apps and examples of classroom activities conducted with various apps. The tables provided in the subsequent sections also provide examples of Common Core State Standards (CCSS) that could be addressed by integrating coding apps.

Creating Rather Than Consuming

Well-designed coding apps encourage users to create digital content rather than just consume it. They provide an interface in which users must think creatively, reason systematically, and often work collaboratively to program their own interactive stories, games, and animations. Examples of apps that are useful for this type of creation are Scratch (scratch.mit.edu) and ScratchJr (www.scratchjr.org). Scratch is a free program targeted at children ages 8–16 that allows users to program their own interactive stories, games, and animations. It has a rich online community with resources for parents and teachers and, in this online space, users can upload creations to be remixed by others. ScratchJr is an app targeted at younger children with the same goal of creating interactive stories and games. Although the app is developmentally appropriate for younger children, we believe that it is also appropriate for older students who have less experience with programming applications.

ScratchJr has an icon-based interface with only a small amount of alphabetic text, so it is not necessary for the user to be able to read alphabetic text to navigate the app. Thus, the app is appropriate for younger users, English learners, and students who need to develop their visual literacy skills. The purpose and functions of the app are introduced with a video that can be viewed by clicking on a question mark icon on the main screen. The video has no written or spoken instructions; therefore, students must have strong viewing comprehension skills to understand the instructions in the video. Accordingly, this video provides an opportunity to focus on the Common Core State Standard requiring students to "summarize a written text read aloud or information presented in diverse media and formats, including visually, quantitatively, and orally" (CCSS.ELA-LITERACY.SL.5.2). Teachers could use this video as an opportunity to teach students how to both understand and create visual representations of information.

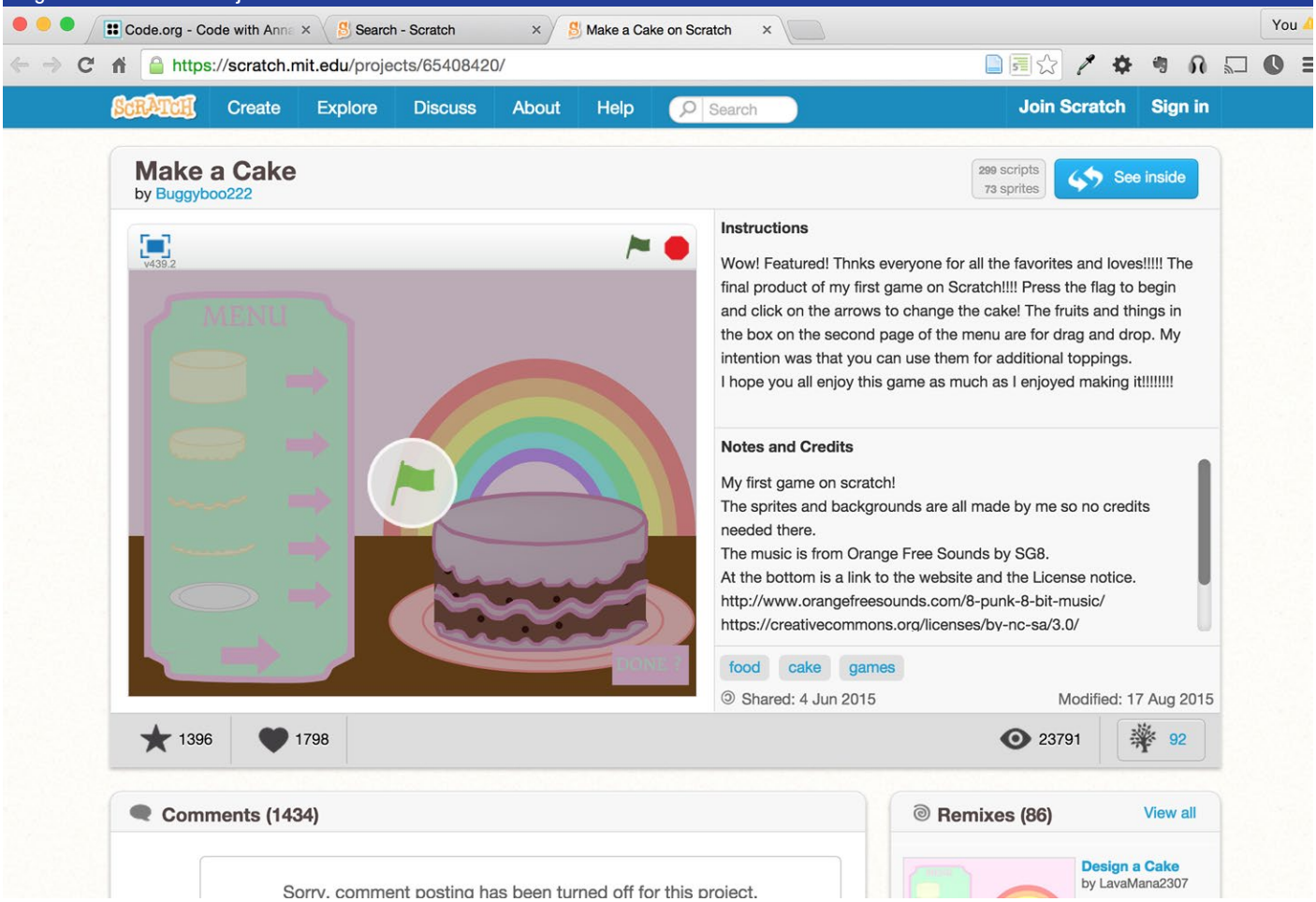
As it relates to literacy instruction, students could use Scratch to discover the disciplinary skills and literacies involved in various careers and practices. Knobel and Lankshear (2014) argue that "playing with and exploring the affordances of a

"As young children code with ScratchJr, they learn how to create and express themselves."

technology or online space balanced with serious work is a key element in learning to 'be' someone, like a machinima artist, a games designer, a video editor, etc." (p. 100). Accordingly, Scratch can be used as a tool for students to explore what it is like to be a computer programmer, game designer, digital artist, illustrator, writer, and so on. Students will simultaneously have the opportunity to discover the disciplinary skills and literacies involved in such careers and practices and use their literacy skills to create and compose digital products. For example, students could use Scratch to collaboratively design and create a game to explore the disciplinary skills of digital game designers. To design a game with Scratch, students must employ a wide range of skills and practices that include but are not limited to (1) inventing and scripting a creative idea for an original game; (2) creating or selecting illustrations, images, colors, and sounds for the game; (3) using programming logic to create the code that will make the game work; and (4) writing directions that tell how to play the game. During this process, students are not only exploring the affordances of a technology, learning how to code, and learning the disciplinary practices of a game designer; they are also employing a wide range of literacy skills that address multiple English Language Arts standards, such as the following anchor standards: (1) CCSS.ELA-LITERACY.CCRA.W.4: "Produce clear and coherent writing in which the development, organization, and style are appropriate to task, purpose, and audience"; (2) CCSS.ELA-LITERACY.CCRA.SL.1: "Prepare for and participate effectively in a range of conversations and collaborations with diverse partners, building on others' ideas and expressing their own clearly and persuasively"; and (3) CCSS.ELA-LITERACY.CCRA.L.6: "Acquire and use accurately a range of general academic and domain-specific words and phrases sufficient for reading, writing, speaking, and listening at the college and career readiness level."

Further, the Scratch website features a rotating selection of student-created projects. One popular project at the time

Figure 2 Featured Project on the Scratch Website



of this writing is Make a Cake, in which students can add toppings to create their own unique virtual cake (see Figure 2). Although the project was only shared to the Scratch site a month prior to the date of this writing, it had received 18,841 views and had been remixed 67 times—that is, at least 67 other users viewed and modified the code for this game and shared their modified versions to the Scratch website. Remixing is a popular feature of the Scratch website that allows users to learn from, experiment with, and add onto the work of others. Thus, the Scratch website provides space for a collaborative community of users to gain experience and skills through existing projects of interest. Additionally, users can comment on projects posted to the Scratch site and suggest ways to improve the project concept or the code for the project. For example, Make a Cake has received 1,257 comments as of this writing. One of those comments states, "Awesome! Why don't you try and add something in that allows you to delete a topping if you change your mind?" In the same vein, the creator of the project can respond to users' comments and even remix his or her own game.

Through the project creation and interaction facilitated by Scratch, students employ a range of skills and have opportunities to develop collaboration and communication skills. Thus, by engaging in what may seem to be only a coding activity, students are also learning general and disciplinary literacy skills, technology skills, and collaboration skills, and they are able to choose projects that interest them and that may encourage them to continue creating outside of the classroom.

Although there are many more ways that Scratch and ScratchJr could be integrated into classroom literacy instruction to teach Common Core English Language Arts standards, we provide some simple ideas for getting started with ScratchJr in Table 1. Although we use fourth grade as an example, these ideas could be modified for nearly any elementary grade level.

Table 1 Ideas for Integrating ScratchJr Into Literacy Instruction

English Language Arts Standards Addressed	Integration Idea
CSS.ELA-LITERACY.RI.4.2 "Determine the main idea of a text and explain how it is supported by key details; summarize the text." CSS.ELA-LITERACY.SL.4.5 "Add audio recordings and visual displays to presentations when appropriate to enhance the development of main ideas or themes."	Have students create a character or scene illustrating the main idea of a text (or portion of a text) and use the audio function to explain the details and summarize the text. Have students present their creations to their peers.
CCSS.ELA-LITERACY.RL.4.3 "Describe in depth a character, setting, or event in a story or drama, drawing on specific details in the text (e.g., a character's thoughts, words, or actions)."	Have students choose existing characters and scenes in the ScratchJr app and add code to modify and animate them to present characters and events from a story they have read. Students can use the audio recording feature to describe the scene and include characters' thoughts or specific details from the story.
CCSS.ELA-LITERACY.RI.4.4 "Determine the meaning of general academic and domain-specific words or phrases in a text relevant to a grade 4 topic or subject area."	Students can create scenes to illustrate the meaning of domain-specific words and phrases. Students can use the animation commands to include actions to better illustrate words and show their understanding of words. Students can also use the text insertion feature to label words and phrases.
CCSS.ELA-LITERACY.RI.4.5 "Describe the overall structure (e.g., chronology, comparison, cause/effect, problem/solution) of events, ideas, concepts, or information in a text or part of a text."	Students can create multiple scenes or events and program them to play in chronological order to show a sequence of events, to show a problem followed by a solution, or to illustrate cause-and-effect relationships among events or concepts in a text.
CCSS.ELA-LITERACY.RI.4.6 "Compare and contrast a firsthand and secondhand account of the same event or topic; describe the differences in focus and the information provided."	Students can create a scene from a text, such as a newspaper article, and create multiple characters to provide different accounts of the same event or topic. Students can then describe how the characters' accounts differed and why.
CCSS.ELA-LITERACY.W.4.3.a "Orient the reader by establishing a situation and introducing a narrator and/or characters; organize an event sequence that unfolds naturally."	Have students brainstorm and draft story characters and events by creating characters and scenes using ScratchJr. Students can use these creations as a basis for writing a corresponding story.

Understanding If–Then Relationships, Navigating Informational Text, and Sequencing Events

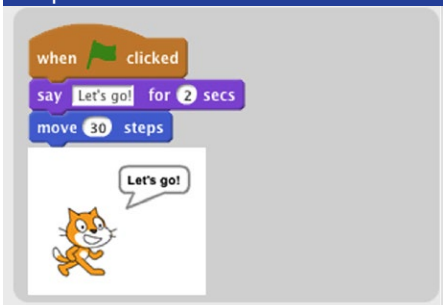
Coding apps also support students' understanding of if–then relationships, learning how to navigate informational text, and experiencing the importance of sequencing events correctly. For example, many coding apps allow users to use if–then commands to control animations they create; the creator may specify a sequence such as "if the start button is clicked, then the animation starts." This programming sequence would ensure that the animation would not start until the start button was clicked.

Similarly, most programming apps targeted at students use a visual programming language that requires users to connect code blocks together to create commands. All of these functions

require users to navigate informational text that is presented as images and icons, written text, audio, and video and that often requires users to learn domain-specific vocabulary. Figure 3 provides an example from Scratch that shows a logical sequence of coding blocks that could be linked together. This sequence of blocks commands that when the green flag is clicked, the character will say "let's go" for 2 seconds, then move 30 steps. There are endless configurations that could be put together to make the character do what the user desires. The important idea is that the user must carefully think through the correct sequence of events and drag the blocks together correctly to create the desired effect. The command blocks are simple to read and understand, but careful computational thinking is required to sequence commands in an effective way. This type of thinking and creating provides students with an introduction to the type of disciplinary writing necessary for computer programming. It could also serve as an introduction to the idea that different types of writing are needed for different situations.

Another app that can be used to promote understanding of if–then relationships, navigation of informational texts, and understanding of logical sequencing is Tynker (www.tynker.com) for tablets. Tynker is a free app that is more advanced than Scratch and ScratchJr and would thus be best used once students have been introduced to more basic apps or have had other programming experiences. The interface is similar to ScratchJr but is less simplistic and does not include as many detailed tutorials. Tynker has options to play preprogrammed games that teach computational thinking and basic coding concepts or to create new projects and games using preloaded templates and characters. The "create" option encourages users to choose and create characters and scenes and thus has a natural connection to the English Language Arts standards. For example, using the "create" option, standard CCSS.ELA-LITERACY.W.3.3 could be addressed (the full standard is listed in Table 2, where we present some additional ideas for integrating Tynker into literacy instruction). This standard asks that students establish characters, introduce a sequence of events, and use dialogue and descriptions to develop events. Scenes that students create with Tynker could be used as visual outlines for a story that students plan to write. For example, Figure 4 shows a screenshot of a visual summary of a student's story in which the main character is a budding actress with an overactive

Figure 3 Coding Blocks Showing Sequence of Events for a Character



You can type in any words to say.

The number of seconds tells the speech bubble how long to show. The script waits that long before continuing.

Table 2 Integrating Tynker Into Literacy Instruction

English Language Arts Standards Addressed	Integration Idea
CCSS.ELA-LITERACY.RI.3.3 "Describe the relationship between a series of historical events, scientific ideas or concepts, or steps in technical procedures in a text, using language that pertains to time, sequence, and cause/effect."	To create games and change the properties of preloaded characters and game templates, players must plan and code a logical sequence of steps to program the game. Tynker provides an excellent opportunity to introduce students to technical writing and language related to sequence or cause and effect. After creating the correct code to program their own game, students can discuss how the technical procedures they created led to their completed product. Students can also describe how the code they created led to a specific sequence of events and created cause-and-effect relationships.
CCSS.ELA-LITERACY.RI.3.5 "Use text features and search tools (e.g., key words, sidebars, hyperlinks) to locate information relevant to a given topic efficiently."	Tynker includes many features of traditional informational text. Users must navigate keywords and sidebars to discover the functions and features of the app. This app could be used to help students understand how keywords and sidebars provide access to information that is needed.
CCSS.ELA-LITERACY.W.3.3 W.3.3.a: "Establish a situation and introduce a narrator and/or characters; organize an event sequence that unfolds naturally." W.3.3.b: "Use dialogue and descriptions of actions, thoughts, and feelings to develop experiences and events or show the response of characters to situations." W.3.3.c: "Write narratives to develop real or imagined experiences or events using effective technique, descriptive details, and clear event sequences."	The Create function in Tynker is all about letting students create scenes, characters, and events, then using coding commands to create a desired sequence of events. Thus, this app would be a great complement to a writing activity in which students are creating characters and stories. Students could create characters and program their actions, then use those characters and events as the basis for their writing.

Figure 4 Creating a Visual Story Outline With Tynker

Figure 5 Tynker Interface and Coding Commands

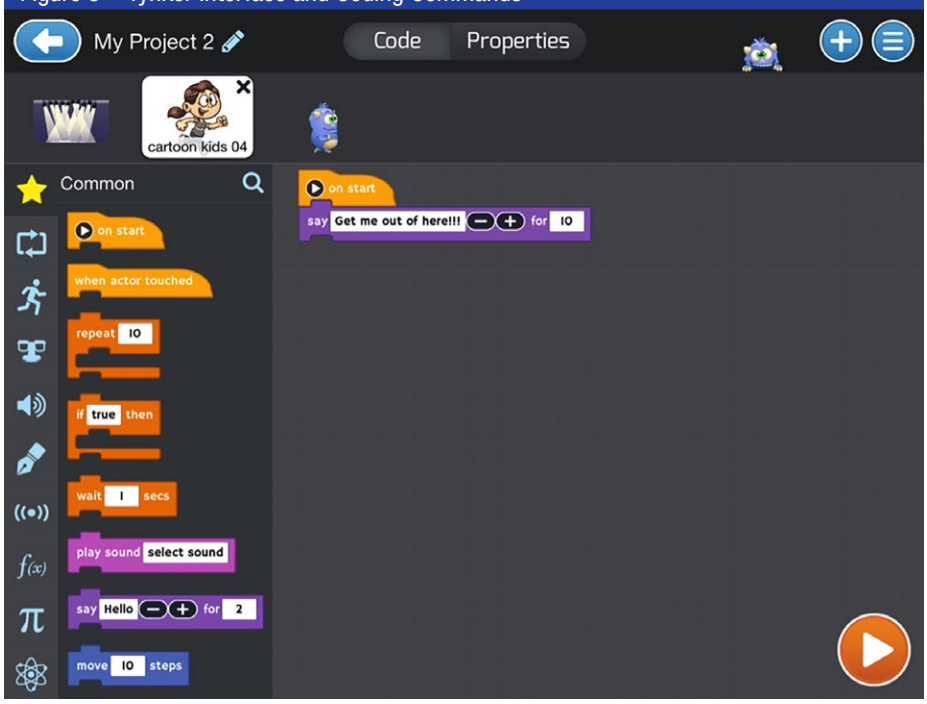
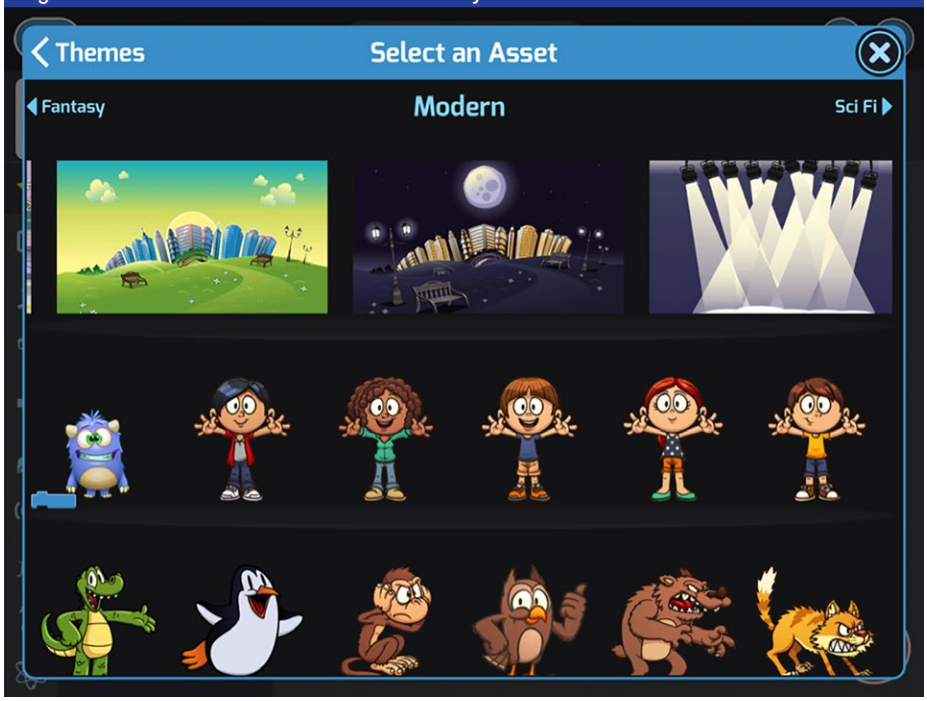


Figure 6 Scene and Character Selection in Tynker



imagination who has been dreaming of her big day on the stage but is frightened away when she imagines that a purple monster is trying to sabotage her debut.This simple scene was created in Tynker by selecting preloaded backgrounds and characters, which served as story starters for the author by providing ideas about a story that could be created. (Tynker also provides an option to create scenes and characters from scratch without using preloaded templates.) The coding blocks were then sequenced to program the female character to say "get me out of here." Figure 5 shows the Tynker interface and the programming blocks used for this scene. Figure 6 shows how scenes and characters were selected.

Logic, Reasoning, and Problem Solving

Coding apps can also help students practice logic, reasoning and problem solving to create or manipulate digital content through the use of computational thinking. In this process, students are "identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources" (International Society for Technology in Education, 2011). One example of an app that is helpful for teaching computational thinking and uses a range of literacy skills is My Robot Friend.

In My Robot Friend, a game produced by LeapFrog that is aimed at teaching computational thinking,

Figure 7 Logic and Reasoning in My Robot Friend

players are instructed to make a robot move in a directed way. The game is highly engaging, progresses through increasingly complex levels, and provides detailed instructions that help players progressively build programming skills. My Robot Friend primarily instructs players through visual images, but audio instructions are also provided throughout to introduce concepts and build the game narrative as the programming requirements, and correspondingly the story plot, become more complex. Accordingly, students must employ their visual and listening comprehension skills to navigate the game. Teachers can use this engaging game to help students learn programming logic and build a variety of literacy skills such as (1) learning to summarize visual and audio information by requiring students to summarize the game events and plot; (2) reporting on a topic and providing descriptive details to support main ideas and themes by having students interpret and orally report on the plot and themes presented in the game and present an opinion of the game; (3) comparing and contrasting characters and events by comparing the good and bad characters in the game, thus using the game as an introduction to help students learn about character description; and (4) analyzing how visual and multimedia elements contribute to meaning and tone by having students analyze how the images, sounds, and text presented in the game work together to convey the plot. Figure 7 shows a My Robot Friend scene in which Fat Cat (sitting on a perch on the right side of the image), the antagonist in the story of the game, is providing verbal instructions to the player. The player must use logic and reasoning to figure out how to sequence the cards that move the robot forward to collect the treasure. The player must also problem-solve to determine how to use the cards provided in a way that will help the robot avoid the guard that is protecting the treasure. Table 3 provides a sampling of ideas for using My Robot Friend to connect to fifth-grade literacy standards.

Conclusion

Because coding literacy is a relatively new concept for many educators, it may be difficult to understand its relationship

Table 3 Integrating My Robot Friend Into Literacy Instruction

English Language Arts Standards Addressed	Integration Idea
CCSS.ELA-LITERACY.RL.5.3 "Compare and contrast two or more characters, settings, or events in a story or drama, drawing on specific details in the text (e.g., how characters interact)."	Teachers can use this game as a way to help students learn to describe characters in a story by comparing and contrasting the robot friend, the main character, with Fat Cat, the "bad guy" in the game. Fat Cat has strong character traits that are presented through written dialogue, which makes him ideal for contrasting with the main character. Thus, My Robot Friend can act as an introduction to character description that can be applied to written text. However, it should be noted that Fat Cat is not introduced until the second level of the game. Students learn basic concepts in the first level of the game before moving on to the plot-driven portion.
CCSS.ELA-LITERACY.RL.5.7 "Analyze how visual and multimedia elements contribute to the meaning, tone, or beauty of a text (e.g., graphic novel, multimedia presentation of fiction, folktale, myth, poem)."	My Robot Friend is presented mostly visually, with some supporting text as needed. Therefore, most of the information about the game, including the tone and plot, is presented visually. This game could be used as an introduction to understanding the role of images in both printed and digital texts. It has many visual and audio elements that contribute to creating the mood, which in turn contributes to understanding the overall plot of the game.
CCSS.ELA-LITERACY.SL.5.2 "Summarize a written text read aloud or information presented in diverse media and formats, including visually, quantitatively, and orally."	Teachers can use My Robot Friend as a way to teach students how to interpret and summarize information that is presented visually and orally. Students can summarize the game events and plot.
CCSS.ELA-LITERACY.SL.5.4 "Report on a topic or text or present an opinion, sequencing ideas logically and using appropriate facts and relevant, descriptive details to support main ideas or themes; speak clearly at an understandable pace."	Students can interpret and orally report on the plot and themes presented in the game and to present an opinion of the game. Students should discuss the sequence of events in My Robot Friend and provide descriptive details to describe the main idea of the game. Teachers can use this game as a way to introduce concepts related to determining main ideas before asking students to apply these concepts to written text.

"Literacy learning may need to look different when you integrate this type of instruction."

to literacy and classroom settings. We believe that this article illuminates several important reasons for developing coding literacy and how this type of literacy can support and enhance the development of traditional literacy skills. We encourage teachers to use the integration ideas presented in this article to select relevant high-quality coding apps for integration into their own classrooms to teach the Common Core English Language Arts Standards. However, there are a few things that teachers should consider as they attempt to integrate coding apps and teach coding literacy. Through their Technology Integration Planning Cycle for Literacy and Language Arts, Hutchison and Woodward (2014) suggest that teachers should consider several aspects of planning or instruction to plan lessons that integrate digital technology. Based on their approach, we suggest the following considerations for teachers who wish to integrate coding apps into their classrooms:

1. **Instructional Goals.** How do these apps connect to the instructional goals you have set for your students? Be certain that there is a clear connection between your instructional goals and the apps you will be using to ensure that your focus remains on literacy learning and not only on using technology.
2. **Instructional Approach.** Literacy learning may need to look different when you integrate this type of instruction into your classroom. Consider in advance the types of groupings that might work best for your students, the learning outcomes that you want to achieve, the length of lesson that will be necessary to integrate these apps, and the overall pedagogical approaches that will best support your learners.
3. **How These Apps and Teaching Approaches Can Benefit Students.** There are often complications when teachers try to integrate new apps and instructional approaches into the classroom. However, we encourage teachers to consider in advance the many ways that integrating these apps and these new approaches to teaching literacy may benefit students. Consider also how your approach aligns with the technology goals set in the CCSS and 21st-century literacy standards (National Council of Teachers of English, 2008). Considering these benefits in advance may help you to persevere when you encounter barriers.
4. **Possible Barriers.** Try out the apps that you are planning to use in the way that you are planning to use them for instruction to find any potential barriers that your students may face. For example, an app may require a login or may not have an obvious way to save student work. Once you have identified some of these potential issues, consider how you can overcome these barriers, such as creating user names in advance to share with your students and save time in class or finding an alternate way for students to share their work.
5. **Instructional Implications and Classroom Context.** It will be important to consider how your typical classroom setup and instruction may need to change as a result of integrating digital technology into your instruction in this way. For example, does the physical space in the classroom work? Do you need a new way for students to turn in their work? Have you considered how you will assess students' learning in this context? Considering these potential changes in advance will help you design your instruction to best support the outcomes you desire.
6. **Time for Reflection.** Finally, it is important to be reflective as a teacher, but reflection can be particularly beneficial when you are integrating new tools, apps, or approaches into your classroom. Take time to consider the extent to which students met the instructional goal of your lesson. Did the technology detract from or support student learning? Consider how you can adjust your instruction in the future to best support learning.

"We encourage teachers to consider in advance the many ways that integrating these apps and these new approaches to teaching literacy may benefit students."

Although integrating new digital tools into instruction can be daunting, it is critical that teachers give students the opportunities that they need to be fully literate, which includes providing them with opportunities to learn with digital technology (International Reading Association, 2009). In this article, we have provided support for teachers to integrate digital technology into their classroom to help students become 21st-century literate. As you try out these new approaches to literacy instruction, we encourage you to share your ideas and reflections with other teachers through networks such as Twitter or other online learning communities in which you may be involved. By using the digital tools that you wish to teach your students to become part of a collaborative community of learners, you can learn from our own experiences and from others. That is, after all, a part of what it means to be literate in the 21st century.

TAKE ACTION!

- Review the integration ideas presented in this article, and select one app for inclusion in your classroom.
- Download and play the app, making sure to explore all buttons and features. If this is an app with levels, play beyond the level your students may reach in class.
- In preparation for the activity, consider asking for parent or teacher volunteers to assist with student questions.
- During the activity, pay close attention to student behaviors. For example, do students lose attention quickly? Do they struggle with certain elements of the app, and how do they overcome their struggles? In addition, take note of your role in facilitating learning. What questioning strategies encourage students to go deeper? How can you effectively redirect struggling students?

REFERENCES

Code.org. (2014). *2014 annual report*. Retrieved from code.org/about/2014

Coiro, J., Knobel, M., Lankshear, C., & Leu, D.J. (Eds.). (2008). *Handbook of research on new literacies*. Mahwah, NJ: Erlbaum.

Gee, J.P. (2013). *Good video games and good learning: Collected essays on video games, learning, and literacy* (2nd ed.) New York, NY: Peter Lang.

Hutchison, A., & Woodward, L. (2014). A planning cycle for integrating technology into literacy instruction. *The Reading Teacher, 67*(6), 455–464. doi:10.1002/trtr.1225

International Reading Association. (2009). *New literacies and 21st-century technologies: A position statement of the International Reading Association*. Newark, DE: Author.

International Society for Technology in Education. (2011). *Operational definition of computational thinking for K–12 education*. Retrieved from www.iste.org/docs/ct-documents/computational-thinking-operational-definition-flyer.pdf?sfvrsn=2

Johnson, M. (2015, July 14). Should my kid learn to code? [Web log post]. Retrieved from googleforeducation.blogspot.com/2015/07/should-my-kid-learn-to-code.html

Knobel, M., & Lankshear, C. (2014). Studying new literacies. *Journal of Adolescent & Adult Literacy, 58*(2), 97–101. doi:10.1002/jaal.314

Lankshear, C., & Knobel, M. (2007). Sampling "the new" in new literacies. In M. Knobel & C. Lankshear (Eds.), *A new literacies sampler* (pp. 1–24). New York, NY: Peter Lang.

Leu, D., Kinzer, C., Coiro, J., Castek, J., & Henry, L. (2013). New literacies: A dual level theory of the changing nature of literacy, instruction, and assessment. In D.E. Alvermann, N.J. Unrau, & R.B. Ruddell (Eds.), *Theoretical models and processes of reading* (6th ed., pp. 1150–1181). Newark, DE: International Reading Association.

Mechaber, E. (2014, December 10). President Obama is the first president to write a line of code [Web log post]. Retrieved from www.whitehouse.gov/blog/2014/12/10/president-obama-first-president-write-line-code

National Council of Teachers of English. (2008). *The definition of 21st-century literacies*. Retrieved from www.ncte.org/governance/literacies

Orsini, L. (2013, May 31). Why programming is the core skill of the 21st century [Web log post]. Retrieved from readwrite.com/2013/05/31/programming-core-skill-21st-century

Papert, S. (1993). *The children's machine: Rethinking school in the age of the computer*. New York, NY: Basic.

ScratchJr.org. (2015). *What is ScratchJr?* Retrieved from www.scratchjr.org/about.html

Warschauer, M. (2006). Literacy and technology: Bridging the divide. In D. Gibbs & K.-L. Krause (Eds.), *Cyberlines 2.0: Languages and cultures of the Internet* (pp. 163–174). Albert Park, VIC, Australia: James Nicholas.