teach beginning coding WITH SCRATCH™

An **ELEMENTARY** Teacher's Guide

A WORLD OF IDEAS: **SEE ALL THERE IS TO KNOW**

An **ELEMENTARY** Teacher's Guide
prepared by Grant Smith

To complete the projects outlined in this guide, your students will need the following DK publications:

Note to Educators:

Coding with Scratch Workbook, ***Coding in Scratch: Games Workbook***, ***Scratch Challenge Workbook***, ***Coding Projects in Scratch***, and ***Coding Games in Scratch*** serve as workbooks and guides that are useful in helping young learners understand and create simple computer programs. The books contain project walk-throughs that will engage all types of learners. Many projects also include suggested "hacks and tweaks" that are perfect for differentiated learning.

The workbooks (***Coding with Scratch Workbook***, ***Coding in Scratch: Games Workbook***, and ***Scratch Challenge Workbook***) provide a scaffolded learning environment that can be used to build student knowledge. These workbooks also contain short quizzes that can serve as formative or summative assessments.

The larger guides (***Coding Projects in Scratch*** and ***Coding Games in Scratch***) promote student choice by outlining projects in many fields of interest including art, music, games, simulations, and more. These projects are perfect for aligning your lessons to another content area.

Seymour Papert

In Mindstorms (1980), Papert wrote: *"One might say the computer is being used to program the child. In my vision, the child programs the computer, and in doing so, both acquires a sense of mastery over a piece of the most modern and powerful technology and establishes an intense contact with some of the deepest ideas from science, from mathematics, and from the art of intellectual model building."*

Scratch builds on the constructionism learning theory. Seymour Papert developed this theory as a branch of Jean Piaget's description of constructivism. However, the difference is that Papert focused on the social aspects of learning. The idea is that students learn best when they use the knowledge they have to build an artifact and share it. Scratch was intentionally developed not only to provide an open and creative coding sandbox, but also to allow users to build and share projects on a global scale.

Papert explained that "the role of the teacher is to create the conditions for invention rather than provide ready-made knowledge." This teacher guide has been developed with the goal to assist teachers (even those with little to no computer science content knowledge) in effectively creating the ideal "conditions for invention" for their students. At this point you may be asking yourself the following questions:

- ***Will I know the answer to every question that my students will have?***
- ***Will I feel well-rested, prepared, and in control at all times?***
- ***Will every class run without a hitch?***

Let me answer those questions for you:
1) No. 2) You wish. 3) In your dreams!

Will it be worth it? You better believe it!
Now let's make it happen!

GETTING STARTED

All the projects in this teacher guide will be made by your students using Scratch. Scratch was developed by the Lifelong Kindergarten group at the MIT Media Lab. It costs nothing and takes student privacy seriously. Scratch is a coding environment where students snap blocks together to create computer programs. The advantage to learning in a block-based environment is that your students won't have to worry about complicated language syntax. Additionally, Scratch provides a learning environment that is easy to get started with, allows a wide variety of types of projects, and is powerful enough to create fairly complex programs.

Before you get started with your students, you will need to set up Scratch. There are online, offline, and iPad versions of Scratch. To choose which one is the best for your class, you will need to consider the following (and it may be a good idea to get help from your technology department):

- ***Does your school/district have policies against creating online student accounts?***
- ***Are you able to download programs/apps onto your devices?***
- ***What kinds of devices do you have?***

Pages 6–7 of the ***Coding with Scratch Workbook*** provide an overview of how to sign up for or download Scratch. If you only have access to iPads, you can use Pyonkee (based off Scratch 1.4, not officially made by MIT). If your district has policies against students under 13 years old creating online accounts, consider applying for a Scratch Educator account.

Before your first lesson, you should complete the student projects found in this teacher guide. Completing the activities yourself will help you anticipate the needs of your own students. Also, your completed projects will serve as models to show students what they will be making.

INTRODUCTION TO PROGRAMMING:

If you are unfamiliar with computer programming, prepare to introduce coding to your students by reading pages 12–23 of the ***Coding Projects in Scratch*** guide. Understand that an algorithm is simply a list of steps to complete a task and a program is an algorithm that a computer can run.

You can introduce coding to your students by showing one or all of the following videos:

Coding for Kids 1: What Is Computer Coding?
http://bit.ly/2rmKFeV

Coding for Kids 2: How Computer Programs Work
http://bit.ly/2qociHm

Coding for Kids 3: Think Like a Computer
http://bit.ly/2qnZnoL

In **Coding for Kids 3: Think Like a Computer**, the robot has to be given very detailed and clear instructions to successfully serve the food. You can model this with your students by pretending to be a robot. Ask students to "program" you by making a list of instructions for you to follow to complete a task (e.g. walk to the door, go to your desk and pick up a pencil, etc.).

INTRODUCING STUDENTS TO SCRATCH

Next, you will introduce your students to Scratch. You can show the **Computer Coding Games for Kids: Introducing Scratch** video (*http://bit.ly/2pQNKnr*) to give an overview of the environment. Then have students open to pages 8–9 of the Coding with Scratch guide and have them point to each area as you say them (Stage area, Sprite list, Stage info . . .). The goal at this point is not to have the students immediately memorize what each button and block in Scratch does. Rather, they should become familiar with the environment and start to develop a common vocabulary when describing what they are doing in Scratch.

For many teachers, the next natural step would be to teach a concept and give an assignment. However, experienced computer science teachers will tell you that after the introduction, students will be so excited to get started that they most likely won't listen to anything else you have to say. That's why you should jump right in by having students login or open Scratch. Set a timer and tell your students they have *X* minutes to "discover something new." Encourage them to share discoveries with their neighbors and to use the correct names when referring to areas in Scratch. While circulating, ask students to point out something interesting they have discovered. Make sure to model using correct terminology. When the time is up, select students to share projects they made or interesting facts they learned. You may be surprised at what students can make and learn without any instruction from you. Encourage divergent thinking and self-reflection.

At the end of your lesson, congratulate your students for becoming computer programmers! Explain that they have some exciting projects ahead of them and will be learning about how to make their computers do amazing things.

LOOPS

The concept of what a loop is can be very easy to understand. However, knowing when and how to use loops effectively is difficult for young students master. It requires thinking about patterns, decomposing problems, and improving programs iteratively. Help your students by guiding them through the explanation found on pages 16–17 of the ***Coding with Scratch Workbook***. Then, have students work on these simple introduction projects:

Coding with Scratch Workbook—Move It!
(found on pages 12–13)

Coding with Scratch Workbook—Which Way
(found on pages 14–15)

Both of these introductory projects come with assessments in the book. If the students aren't ready to move on, have the students customize their Move It! and Which Way projects to use loops in unique ways. You can also have your students complete the animation tutorial on pages 18–19 of ***Coding with Scratch Workbook***.

As with all main projects, the end product shown in the book should not be where your students finish working. The goal for your class should be that all of your students' projects are unique. Many of the projects in the books have a "hacks and tweaks" section. These suggestions are perfect for helping your students generate ideas on how they can make their programs unique. Requiring uniqueness is also a great way to differentiate learning. For some students, their modifications may be simple, which may be perfect for them. Other students will look forward to pushing themselves as they create something more complex. In the end, every student should feel like they were challenged and like they accomplished something they are proud of.

Main project:

Coding Projects in Scratch—Cat Art (found on pages 26–31)

For the main project, your students will create an art program that can be used to create colorful pictures. This project is fairly simple but you should still be prepared to answer questions about blocks that may be new to students. The Cat Art project has some fun hacks and tweaks for students to make their own project unique.

After the students finish their projects, have them explain what loops do. Spend time reflecting on their work. If you have time, you should also delve deeper into the different types of loops found in Scratch (forever, repeat x, and repeat until x).

EVENTS AND PARALLELISM

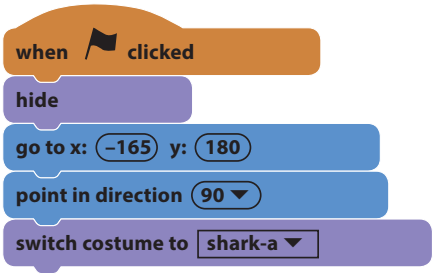
Events and parallelism will bring your students' programs to life. Event blocks trigger scripts to run. Your students have already used events blocks like "when green flag is clicked" and "when space bar is clicked." Parallelism simply means that computers can run multiple scripts at the same time, either independently or in conjunction with each other.

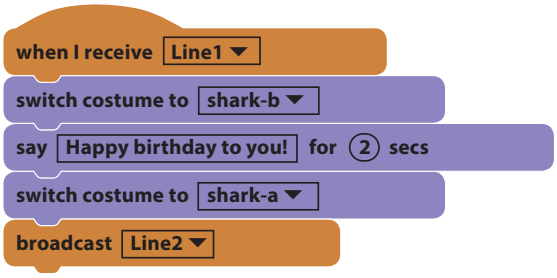
Main project:

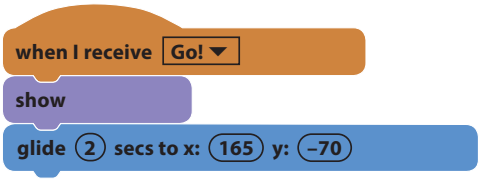
Coding Projects in Scratch—Birthday Card
(found on pages 82–91)

For the main project, students will use events and parallelism to make an interactive birthday card. Events will allow the user to control when chunks of code should run and parallelism will enable multiple sprites to run scripts at the same time. The Birthday Card project has some fun hacks and tweaks for students to make their projects unique.

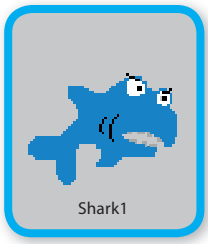
Here are some examples of events in the Birthday Card project:

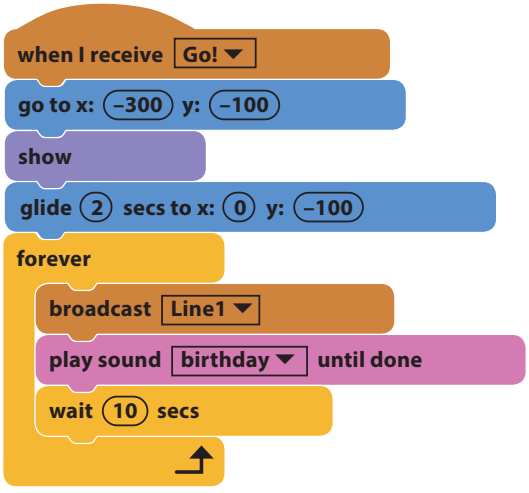


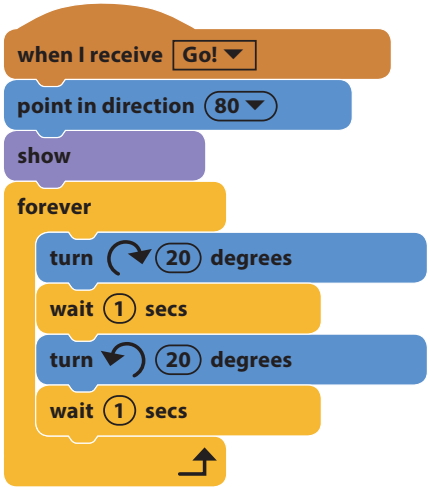


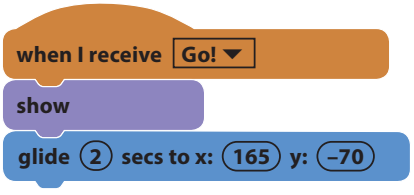


Here are some examples of scripts running in parallel:
When the message "Go!" is broadcast, the Cake, Banner, and Shark1 sprite all run different scripts at the same time.









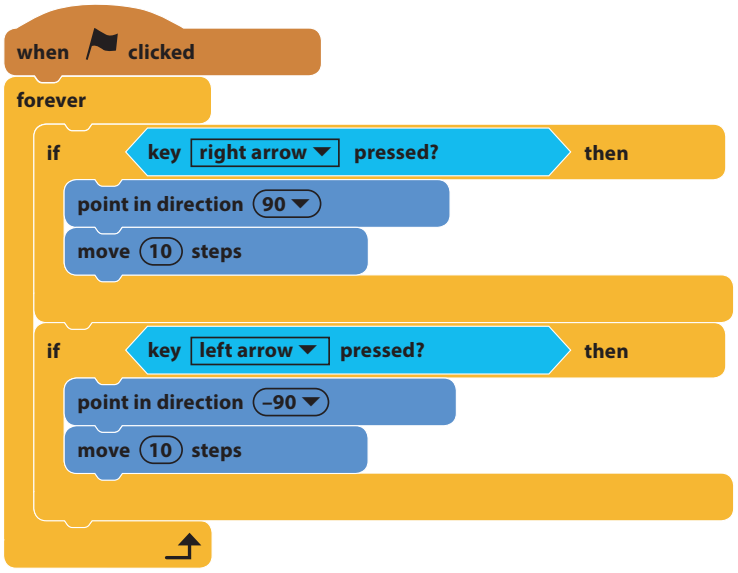
CONDITIONAL STATEMENTS

Conditional statements will make your students' programs smarter. In Scratch programming, conditional statements use the if-then and if-then-else blocks. Have your students work through pages 22–23 of ***Coding with Scratch Workbook*** to get a handle on if-then statements. You may want to have students spend time making their own projects that use if-then blocks. When ready, have your students move on to if-then-else statements by working through pages 30–31 of ***Coding with Scratch Workbook***. Finally, students will complete the main project found below.

Main project:

Coding Projects in Scratch—Dino Dance Party
(found on pages 34–45)

For the Dino Dance Party, students will use loops, events, parallelism, and conditional statements to make animated characters dance on the screen. A small script in the program will check if the left or right arrow key is pressed on the keyboard. If either key is pressed, Dinosaur 3 will move in the corresponding direction. These conditional statements make the program interactive and more fun! Here's what the conditional statements look like in the Dino Dance Party program:



VARIABLES AND OPERATORS

While math operators like addition, subtraction, and division work the same in programming as they do in elementary math, variables are slightly different. Have your students work through pages 24–25 of ***Coding with Scratch Workbook*** to understand variables and pages 26–27 of ***Coding with Scratch Workbook*** to understand how to use math tools in Scratch. A common use for variables in Scratch is to keep track of a player's score in a game. Before jumping to the main project, have your students work on these starter projects:

Coding with Scratch Workbook—A Game: Dragon!
(found on pages 32–33)

Coding Projects in Scratch—Ask Gobo
(found on pages 60–65)

Before moving on to the main projects, give students an opportunity to modify their starter projects to show that they understand how to use variables and math operators in Scratch. When ready, allow students to choose one of the following for their main project:

Coding Games in Scratch—Star Hunter
(found on pages 30–49)

Coding in Scratch: Games Workbook—Ghost Hunt
(found on pages 18–22)

Scratch Challenge Workbook—Keepy-Uppy
(found on pages 16–20)

If students finish early, encourage them to either continue improving their project or choose another project from the list to work on.

CAPSTONE PROJECT

At this point, your students haven't mastered all there is to know about coding, but they have learned enough to make some really neat programs. Give students an opportunity to show off what they know by creating and presenting a capstone project. Have students choose a base project from the list below.

Scratch Challenge Workbook—Memory Master
(found on pages 28–34)

Scratch Challenge Workbook—Monkey Rescue
(found on pages 22–26)

Coding Games in Scratch—Jumpy-Monkey
(found on pages 90–107)

Coding in Scratch: Games Workbook—Fishball
(found on pages 8–16)

Coding in Scratch: Games Workbook—Rapid Reaction
(found on pages 24–28)

Coding in Scratch: Games Workbook—Melon Bounce
(found on pages 30–33)

Coding Projects in Scratch—Animal Race
(found on pages 48–58)

Make sure students modify their end product so that the program is unique. When your students finish, it is a good idea to have them present their projects to their class, parents, or students in other grade levels.

Congratulations! You made it! You have taught your students the basics of coding in Scratch. But don't stop now—keep challenging them! Get your hands on other DK coding books or come up with your own subject-aligned projects. Continue to prepare your pupils for thriving in the twenty-first century.

ARTICLES TO READ

Mitch Resnick: Let's teach kids to code
https://www.ted.com/talks/mitch_resnick_let_s_teach_kids_to_code

Learning to Code Isn't Enough
https://www.edsurge.com/news/2013-05-28-opinion-learning-to-code-isn-t-enough

MORE RESOURCES

Scratch ED
http://scratched.gse.harvard.edu/
An online community for Scratch educators to collaborate and exchange resources.

Downloadable DK Coding Kit
https://www.dk.com/us/explore/education/celebrate-global-scratch-day-with-this-downloadable-computer-coding-kit/

DK 9 Easy Steps Coding Guide
https://www.dk.com/us/explore/education/9-easy-steps-for-teaching-coding-in-the-classroom/

About the Author

Grant Smith is the founder of **Launch CS** (*www.launchcs.com*), the premier provider of K–8 computer science teacher professional development. Grant is also a former K–8 computer science teacher and district administrator. He has led #CSforAll initiatives at multiple school districts across the nation and has developed computer science curricula and standards. He has served on national computer science education teams including the CSTA Standards Review Committee.

CERTIFICATE OF ACHIEVEMENT

THIS CERTIFICATE IS AWARDED TO

COMPUTER CODING EXPERT

FOR DEMONSTRATING AN UNDERSTANDING
OF COMPUTER SCIENCE.

Your Friends at DK

A WORLD OF IDEAS:
SEE ALL THERE IS TO KNOW

www.dk.com