

2022

Elementary Integration Guide

SECOND GRADE



MISSISSIPPI STATE UNIVERSITY™
CENTER FOR CYBER EDUCATION

Acknowledgements

The following people assisted in the development of this integration guide:

Elementary Task Force

Melissa Atkins, Instruction Technologist, Lamar County School District
Kacy Baggett, Teacher, Rankin County School District
Heather Barry-Fenster, Teacher, Lowndes County School District
Brittany Boatman, Teacher, Houston School District
Kimberly Brammer, Teacher, Pascagoula-Gautier School District
Michelle Carter, Teacher, Picayune School District
Jana Chao, Teacher, Clinton Public School District
Lerenda Dixon, Teacher, McComb School District
Samantha Elizondo, Teacher, Tupelo Public School District
Angie Frazier, Teacher, Rankin County School District
Tammy Hale, Teacher, Tate County School District
Kayla Hathcock, Teacher, Amory School District
Ashley Hawkins, Teacher, Biloxi Public School District
Vicky Johnson, Teacher, Franklin County School District
Dinah Lachney, Teacher, Madison County School District
Ashley Matthews, Administrator, Lowndes County School District
Kayla Moore, Teacher, Enterprise School District
Olivia Moore, Teacher, DeSoto County School District
Beth Neese, Teacher, Western Line School District
Hannah Padgett, Teacher, Tate County School District
Dr. Lee Pambianchi, Administrator, Rankin County School District
Kristen Phillips, Teacher, Oxford School District
Brittney Price, Teacher, South Panola School District
Melissa Sundberg, Teacher, Ocean Springs School District
Melissa Tingle, Teacher, Lauderdale School District
Susie Williams, Curriculum Coordinator, Leland School District

Mississippi Department of Education Team

Wendy Clemons, executive director, Office of Secondary Education
Louella Webster, supervisor for computer science/STEM

Center for Cyber Education Team

Shelly Hollis, Director
Lizzie Brandon, Project Manager
Amanda Taylor, Project Manager

Research and Curriculum Unit Team

Brock Turnipseed, Marketing and Communications Manager
Chris McMillen, Communications Coordinator
Heather Craig, Editor
Will Graves, Project Coordinator

Funding for the development of this guide was provided by:



Introduction

In March 2021, The Mississippi Computer Science and Cyber Education Equality Act ([House Bill 633](#)) was passed requiring all districts to offer computer science content and courses by the 2024-2025 school year. The bill allows for a phased-in approach as listed below:

- 2022-2023: All middle schools offer at least one (1) course in computer science, and 50% of elementary schools offer a minimum of one (1) hour of instruction in computer science each week at each grade level.
- 2023-2024: All elementary schools offer a minimum of one (1) hour of instruction in computer science each week at each grade level, and 50% of high schools offer at least one (1) course in computer science.
- 2024-2025: All schools will offer instruction in computer science.

To make the integration of computer science content as seamless as possible for elementary teachers, a task force of elementary teachers, principals, the Mississippi Department of Education, and the Mississippi State University Center for Cyber Education was formed to write an integration guide for each grade level, kindergarten through fifth grade. These guides provide plans for a minimum of 40, 60-minute lessons covering six computer science topics: coding, robotics, digital literacy, digital citizenship, keyboarding, and unplugged activities.

Each guide contains a breakdown of content by integrated subjects, content by computer science topics, and a calendar/pacing guide. Teachers may choose to start at the beginning and teach each lesson once a week in chronological order or teach the lesson that integrates with another core subject topic at a more relevant time. In addition to a lesson overview and links to required resources, each lesson plan maps to a Mississippi Computer Science Standard and a core subject area standard. A suggestion on how to break the lesson into smaller segments to be covered throughout the week is also provided in the "Time needed" section.

There are several resources available in each integration guide. Some may require the creation of accounts, but all resources referenced are free. The pacing guide notes lessons requiring account creation so teachers can plan ahead. A list of sites used is provided for technology departments to whitelist or unblock. All resources may be used on any internet-capable device, including Chromebooks and tablets.

Resources	
Computing resources	<ul style="list-style-type: none"> • Code.org CS Fundamentals <ul style="list-style-type: none"> ◦ 2nd Grade: Course C • Common Sense Digital Media • Kodable • Scratch, Jr.
CS4MS website materials	<ul style="list-style-type: none"> • 2018 Mississippi Computer Science Standards • CS4MS Website
Mouse practice	<ul style="list-style-type: none"> • Alphabetical Order • Mouse Practice <ul style="list-style-type: none"> ◦ Apple Catch ◦ Coyote Concentration (card matching game) ◦ Desert Dive ◦ Frost Bite ◦ Helipopper ◦ Penguin Drop ◦ Pickle Pop ◦ Pig Pile ◦ Simon Sees
Keyboard practice	<ul style="list-style-type: none"> • Astro Bubbles Keyboard Practice • Big Brown Bear • Read Today <p>Unplugged:</p> <ul style="list-style-type: none"> • Keyboard Callout <ul style="list-style-type: none"> ◦ Paper keyboard: Using a paper keyboard, the teacher will call out letters, numbers, symbols, and/or words for students to “type” on their keyboard. ◦ Computer with no internet: The teacher will call out letters, numbers, symbols, and/or words. Students will use their keyboard to type into a blank document on their computer/tablet. • Keyboard Bingo <ul style="list-style-type: none"> ◦ Preparation: The teacher will print squares with letters, numbers, and symbols (4-5 of each letter, 1-2 of each number/symbol). The teacher will cut out and laminate each square, then use a piece of tape or glue to adhere a magnet to each square. ◦ The teacher will project a keyboard onto a smartboard. ◦ The teacher will call out letters, numbers, symbols, or words for students to find using their preprinted squares. ◦ Students will raise their hands if they have the key that the teacher calls out. The teacher will choose a student to place their key on the board.
Teacher / student accounts	<ul style="list-style-type: none"> • Code.org • Common Sense Digital Media • Kodable • Scratch, Jr.
For help with this guide	<ul style="list-style-type: none"> • Contact Mississippi State University's Center for Cyber Education: www.tinyurl.com/ccehelpdesk

Contents by Integrated Subjects

English

- **Week 1:** L.2.5—Vocabulary acquisition and use
- **Week 2:** L.2.5—Vocabulary acquisition and use
- **Week 3:** L.2.5—Vocabulary acquisition and use
- **Week 11:** SL.2.1—Comprehension/collaboration
- **Week 13:** L.2.1, L.2.2—Grammar usage, Capitalization/punctuation/spelling
- **Week 14:** SL.2.1—Collaborative conversations with diverse partners
- **Week 18:** W.2.2—Informative/explanatory text
- **Week 19:** W.2.1—Opinion pieces
- **Week 21:** W.2.2, W.2.5—Informative/explanatory text, Revision/editing
- **Week 22:** W.2.1—Opinion pieces
- **Week 23:** W.2.1—Opinion pieces
- **Week 24:** SL.2.1—Collaborative conversations with diverse partners
- **Week 25:** SL.2.1—Collaborative conversations with diverse partners
- **Week 27:** SL.2.1—Collaborative conversations with diverse partners
- **Week 29:** SL.2.6—Complete sentences
- **Week 30:** SL.2.3—Ask/answer questions
- **Week 32:** SL.2.1, SL.2.2, SL.2.3—Collaborative conversations with diverse partners, Recount/describe key details, Ask/answer questions
- **Week 35:** RL.2.1, SL.2.1—Who/what/when/where, Collaborative conversations with diverse partners
- **Week 26:** RL.2.7—Characters/setting/plot
- **Week 38:** SL.2.1, SL.2.2, SL.2.3—Collaborative conversations with diverse partners, Recount/describe key details, Ask/answer questions
- **Week 39:** RF.2.3—Grade-level phonics

Math

- **Week 11:** 2.MD.9—Measurement representation and data interpretation
- **Week 12:** 2.OA.2, 2.NBT.4—Add/subtract within 20 using mental math, Place values
- **Week 16:** 2.G.1—Shapes and their attributes
- **Week 17:** 2.MD.3—Measure and estimate lengths
- **Week 18:** 2.OA.2—Add/subtract within 20 using mental math
- **Week 19:** 2.G.1—Shapes and their attributes
- **Week 21:** 2.OA.2—Add/subtract within 20 using mental math
- **Week 22:** 2.G.1—Shapes and their attributes
- **Week 23:** 2.G.1—Shapes and their attributes
- **Week 26:** 2.MD.10—Bar/picture graph
- **Week 28:** 2.MD.10—Bar/picture graph
- **Week 33:** 2.G.2—Reason with shapes and their attributes
- **Week 34:** 2.G.2—Reason with shapes and their attributes
- **Week 37:** 2.OA.2—Add/subtract within 20 using mental math
- **Week 40:** 2.G.1—Shapes and their attributes

Science

- **Week 17:** E.2.10.1, E.2.10.4, E.2.10.5—Renewable/nonrenewable resources, Soil erosion, Prevent/repair soil erosion
- **Week 21:** P.2.6.1—Push, pull, friction
- **Week 38:** E.2.10.1—Renewable/nonrenewable resources

Social Studies

- **Week 4:** CI.2.2—Rights/responsibilities of community members
- **Week 5:** CI.2.2—Rights/responsibilities of community members
- **Week 6:** CI.2.2—Rights/responsibilities of community members
- **Week 7:** CI.2.2—Rights/responsibilities of community members
- **Week 8:** CI.2.2—Rights/responsibilities of community members
- **Week 9:** CI.2.2—Rights/responsibilities of community members
- **Week 10:** CI.2.2—Rights/responsibilities of community members
- **Week 15:** G.2.1.1—Map skills
- **Week 20:** E.2.1.1—Identify consumers/producers
- **Week 34:** G.2.3.2—Cardinal/immediate directions.

Contents by Topics

Coding

- Week 12
- Week 13
- Week 14
- Week 15
- Week 16
- Week 17
- Week 18
- Week 19
- Week 20
- Week 21
- Week 22
- Week 23
- Week 24
- Week 25
- Week 26
- Week 27
- Week 28
- Week 29
- Week 30
- Week 31
- Week 32
- Week 33
- Week 34
- Week 35
- Week 36
- Week 37
- Week 38

Digital Citizenship

- Week 4
- Week 5
- Week 6
- Week 7
- Week 8
- Week 9
- Week 10

Digital Literacy

- Week 1
- Week 2

Keyboarding

- Week 3

Robotics

- Week 11
- Week 39
- Week 40

Unplugged

- Week 11
- Week 13
- Week 18
- Week 29
- Week 31
- Week 32
- Week 34
- Week 35
- Week 39
- Week 40

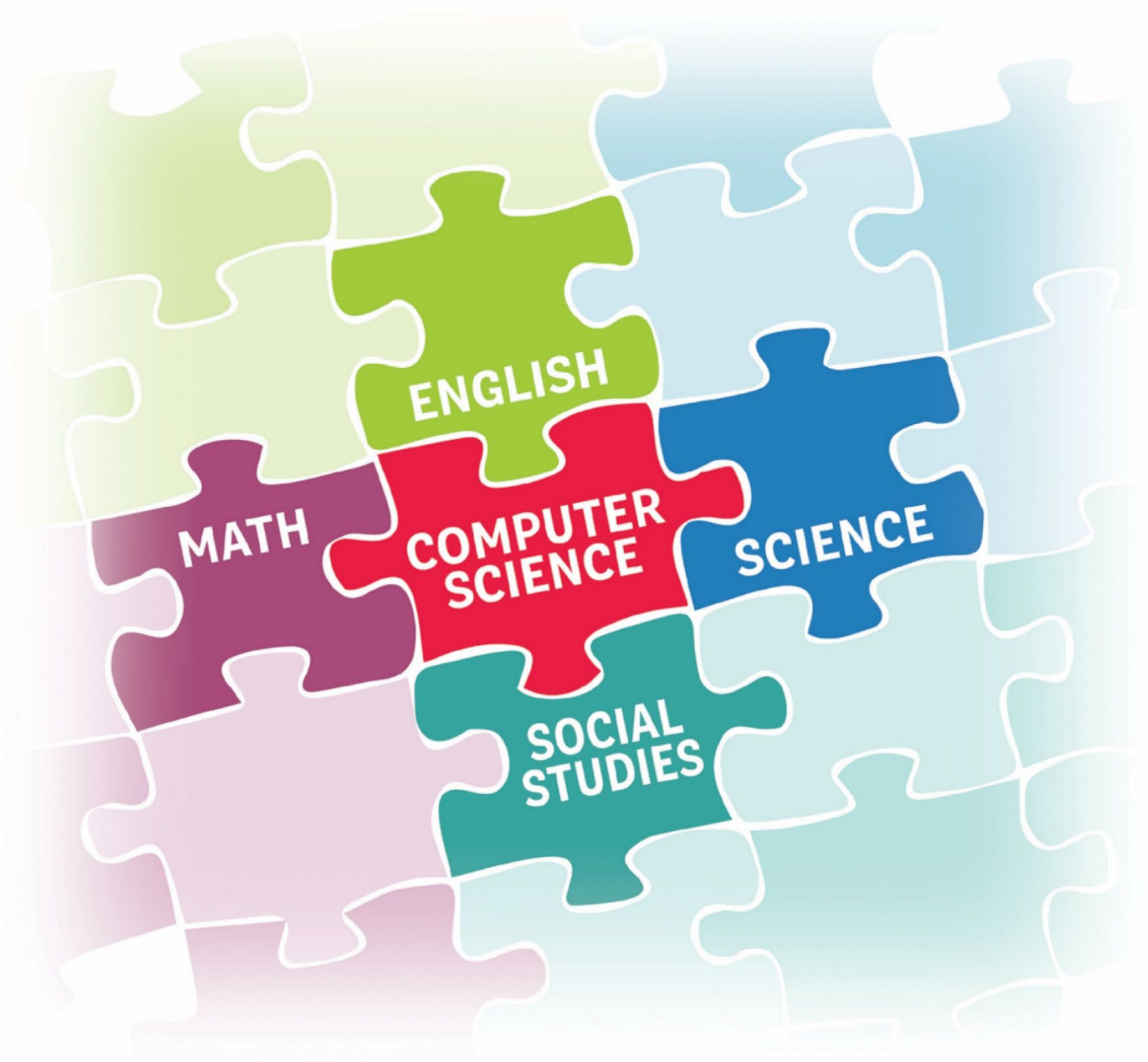
Calendar/ Pacing Per Week:

→ Teachers will need to create a **FREE teacher and/or student account** (see notes section of lesson.)

Week	Title	Topics	CS Standard	Subject Integrated	Core Standard
1	What is a Computer?	Digital Literacy	CS.1A.2	ELA	L.2.5
2	Mouse Practice	Digital Literacy	CS.1A.2	ELA	L.2.5
3	Introduction to Keyboarding	Keyboarding	CS.1A.2	ELA	L.2.5
4	We, the Digital Citizens → Account creation needed	Digital Citizenship	IC.1A.1 IC.1A.2	Social Studies	CI.2.2
5	Device-Free Moments	Digital Citizenship	IC.1A.1 IC.1A.2	Social Studies	CI.2.2
6	That's Private!	Digital Citizenship	IC.1A.3	Social Studies	CI.2.2
7	Digital Trails	Digital Citizenship	IC.1A.3	Social Studies	CI.2.2
8	Who is in Your Online Community?	Digital Citizenship	IC.1A.1 IC.1A.2	Social Studies	CI.2.2 CR.2.1.1
9	Let's Give Credit	Digital Citizenship	IC.1A.2	Social Studies	CI.2.2
10	Code.org: Putting a STOP to Online Meanness → Account creation needed	Digital Citizenship	IC.1A.2	Social Studies	CI.2.2 CR.2.3.1
11	Code.org: My Robotic Friends Jr.	Robotics Unplugged	AP.1A.2	ELA Math	SL.2.1 2.MD.9
12	Code.org: Programming With Angry Birds	Coding	AP.1A.2 AP.1A.4	ELA Math	L.2.1 L.2.2 2.OA.2 2.NBT.4
13	Code.org: Bugs, Bugs, and More Bugs	Coding Unplugged	AP.1A.2 AP.1A.4	ELA Math	L.2.1 L.2.2 2.OA.2 2.NBT.4
14	Code.org: Debugging in Maze	Coding	AP.1A.2 AP.1A.4	ELA	SL.2.1
15	Code.org: Collecting Treasure With Laurel	Coding	AP.1A.2 AP.1A.4	Social Studies	G.2.1.1 G.2.2.3
16	Code.org: Creating Art With Code	Coding	AP.1A.2 AP.1A.4 AP.1A.7	Math	2.G.1
17	Code.org: Shapes and Landscapes	Coding	AP.1A.2 AP.1A.4 AP.1A.7	Math Science	2.MD.3 E.2.10.1 E.2.10.4

					E.2.10.5
18	Code.org: My Loopy Robotic Friends Jr.	Coding Unplugged	AP.1A.2 AP.1A.3 AP.1A.4 AP.1A.7	ELA Math	W.2.2 W.2.5 OA.2
19	Code.org: Loops With Rey and BB-8	Coding	AP.1A.2 AP.1A.3 AP.1A.4 AP.1A.7	ELA Math	W.2.1 2.G.A.1
20	Code.org: Harvesting Crops With Loops	Coding	AP.1A.2 AP.1A.3 AP.1A.4 AP.1A.7	Social Studies	E.2.1
21	Code.org: Harvester's Farm	Coding	AP.1A.5 AP.1A.8	ELA Math Science	W.2.2 W.2.5 2.OA.2 P.2.6.1
22	Code.org: Mini-Project—Sticker Art	Coding	AP.1A.2 AP.1A.3 AP.1A.4 AP.1A.7	ELA Math	W.2.1 2.G.A.1
23	Code.org: Loopy Forms and Their Functions	Coding	AP.1A.2 AP.1A.3 AP.1A.4 AP.1A.5	ELA Math	W.2.1 2.G.A.1
24	Code.org: The Big Event	Coding	AP.1A.2 AP.1A.4	ELA	SL.2.1
25	Code.org: Build a Flappy Game	Coding	AP.1A.2 AP.1A.4	ELA	SL.2.1
26	Code.org: Become a Game Designer	Coding	AP.1A.2 AP.1A.4	Math	2.MD.10
27	Code.org: Mini-Project: Chase Game	Coding	AP.1A.2 AP.1A.4	ELA	SL.2.1
28	Code.org: Picturing Data	Coding	AP.1A.09 AP.1A.11	Math	2.MD.10
29	Code.org: Binary Bracelets	Coding Unplugged	AP.1A.09 AP.1A.11	ELA	SL.2.6
30	Code.org: End of Course Project	Coding	AP.1A.10 AP.1A.12 AP.1A.13 AP.1A.15	ELA	SL.2.3
31	Gingerbread Coding	Coding Unplugged	AP.1A.1 AP.1A.3	Social Studies	G.2.3.2
32	Kodable.com: Pizza Party → Account creation needed	Coding	AP.1A.1 AP.1A.4 AP.1A.5	ELA	SL.2.1 SL.2.2 SL.2.3
33	Kodable.com: Magnificent Maze Maker	Coding	AP.1A.1	Math	G.A.2


34	Kodable.com: Magnificent Maze Maker Extended and Unplugged	Coding Unplugged	AP. 1A.1	Math Social Studies	2.G.2 G.2.3.2
35	Kodable.com: Introduction to Coding and Coding Basics Unplugged	Coding Unplugged	AP.1A.1 AP.1A.3 AP.1A.4 AP.1A.7	ELA	RL.2.1 SL.2.1
36	Kodable.com: Build Your Own Fuzz!	Coding	CS.1A.1	ELA	RL.2.7
37	Kodable.com: If Flash, Then Clap	Coding	AP.1A.4	Math	2.OA.2
38	Kodable.com: Beach Cleanup	Coding	AP.1A.3 AP.1A.4 AP.1A.7	ELA Science	SL.2.1 SL.2.2 SL.2.3 E.2.10.1
39	Spelling Robotics	Robotics Unplugged	AP.1A.4 AP.1A.5	ELA	RF.2.3
40	Marching Orders	Robotics Unplugged	AP.1A.4 AP.1A.3 AP.1A.5	Math	2.G.1



Lessons and Activities

SECOND GRADE

Week 1: What is a Computer?

<p>Lesson overview:</p> 	<p>Purpose: In this lesson, your students will learn the basic parts of computers including computer, monitor, desktop tower, keyboard, mouse, laptop, and tablet. This is an introduction to technology that they will use throughout school. Students will become familiar with the terms, complete a matching worksheet, and write definitions for each computer part while the teacher navigates through a PowerPoint presentation.</p> <p>Lesson:</p> <ul style="list-style-type: none">• Warm Up<ul style="list-style-type: none">◦ (Teacher Prompt): What is a Computer for Kids? - Let students give you the answers to what they think a computer is.◦ Watch the "What is a Computer for Kids" video.• Identifying Computer Parts Activity<ul style="list-style-type: none">◦ Identifying Computer Parts PowerPoint: As you are going through the PowerPoint, students will find and match the pictures of the computer part on the Identifying Computer Parts Worksheet. Once they find the picture, students should draw a line from the picture to the correct term. (Please do not rush through the slides because moving slowly allows students to see both the picture and term on the board.)• Using the "Find the Technology" game, have students start with the "review" feature to learn about different devices; then have them click on the "home" symbol to return to the main page and choose "search" to locate devices in the room displayed on the screen.
<p>Lesson links/resources:</p>	<ul style="list-style-type: none">• What is a Computer for Kids?• Identifying Computer Parts PowerPoint• Identifying Computer Parts Worksheet• "Find the Technology" Game• Keyboarding Practice
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none">• Identify parts of the computer <p>Standards:</p> <ul style="list-style-type: none">• CS.1A.2—The student will use appropriate terminology in identifying and describing the function of common physical components of computing systems (hardware).
<p>Time needed:</p>	<p>Total Time: 60 min</p> <ul style="list-style-type: none">• Warm Up 10 min<ul style="list-style-type: none">◦ What is a Computer for Kids? Video◦ Discussion of video• Activity 1 20 min<ul style="list-style-type: none">◦ Identifying Computer Parts worksheet and PowerPoint (matching)◦ Identifying Computer Parts (re-writing terms)• Activity 2 20 min<ul style="list-style-type: none">◦ "Find the Technology" game• Keyboarding 10 min
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none">• Computer• Projector/smartboard with sound• Identifying Computer Parts PowerPoint <p>Students:</p>

	<ul style="list-style-type: none"> • Computer/tablet with internet access • Identifying Computer Parts Worksheet • "Find the Technology" game • Pencil • Computer
Subject integrated:	ELA
Other standards addressed:	L.2.5 —Demonstrate understanding of word relationships and nuances in word meanings.
Vocabulary:	<p><u>Computer</u>: A device for working with information: data, music, videos, sounds</p> <p><u>Desktop</u>: Screen that appears if you are not browsing the internet, reading a file, or playing a game; icons are on this screen</p> <p><u>Keyboard</u>: Where all the letters, numbers and other buttons are located; when you type on it, the symbols appear on the monitor</p> <p><u>Laptop</u>: A small, portable computer</p> <p><u>Monitor</u>: Screen that shows you what you are doing; a viewer that displays what the computer sends to it</p> <p><u>Mouse</u>: A little device you move with your hand, which then moves the cursor on the screen</p> <p><u>Tablet</u>: A small, handheld computer with a touch screen</p>
Notes:	As your students are completing the worksheet, make sure that you are going through the PowerPoint to show the various images/terms. This will help matching.

Week 2: Mouse Practice

Lesson overview:



Purpose:

Computer [Mouse Practice](#) games help beginning computer users learn mouse skills through hand-eye coordination by dragging, dropping, clicking, double-clicking, and scrolling.

- Apple Catch
- Coyote Concentration (card matching game)
- Desert Dive
- Frostbite
- Helipopper
- Penguin Drop
- Pickle Pop
- Pig Pile
- Simon Sees

Lesson:

- Refresher
 - Review the parts of the computer with your students. Once you have gone back over the parts of the computer, show the video "How Do Computers Work?"
- Warm Up
 - Show the "How to Use a Mouse for Kids" video.
- Mouse Practice
 - **Online:** Go to the mouse practice websites (links below). Let the students choose a game to play to practice using the mouse. (If your students are using a laptop or tablet, that's okay! They can still use the practice to operate a trackpad or touchscreen.)
 - **Unplugged:** Mouse Practice Worksheet. Provide students with coloring utensils and the Mouse Worksheet. Guide students through the completion of the worksheet.
- Student Voice
 - "How does a mouse help a computer?"
 - Give students the opportunity to explain, in their own words, how they think a mouse is helpful.

Lesson links/resources:

- [How Do Computers Work?](#)
- [Using Your Computer Mouse](#)
- [Alphabetical Order](#)
- [Mouse Practice](#)
- [Mouse Practice \(unplugged\)](#)

CS standards addressed:

Students will be able to:

- Operate a mouse and keypad
- Students will be able to find and select letters on a keyboard

Standards:

- **CS.1A.2**—Use appropriate terminology in identifying and describing the function of common physical components of computing systems (hardware).

Time needed:

Total Time: 60 min

- Refresher and Warm Up **15 min**
- Mouse Practice **45 min** (can be divided into 15 min sessions across three days this week)

Materials needed:

Teacher:

- Computer

	<ul style="list-style-type: none"> ● Projector/smartboard with sound ● Identifying Computer Parts PowerPoint(for refresher) <p>Students:</p> <ul style="list-style-type: none"> ● Computer/tablet with internet access ● Alphabetical Order ● Mouse Practice ● Mouse Practice (unplugged) ● Pencil ● Markers, crayons, color pencils, etc.
Subject integrated:	ELA
Other standards addressed:	L.2.5 —Demonstrate understanding of word relationships and nuances in word meanings.
Vocabulary:	<p><u>Computer</u>: A device for working with information: data, music, videos, sounds</p> <p><u>Desktop</u>: Screen that appears if you are not browsing the internet, reading a file, or playing a game; icons are on this screen</p> <p><u>Keyboard</u>: Where all the letters, numbers, and other buttons are located; when you type on it, the symbols appear on the monitor</p> <p><u>Laptop</u>: A small, portable computer</p> <p><u>Monitor</u>- Screen that shows you what you are doing; a viewer that displays what the computer sends to it</p> <p><u>Mouse</u>- A little device you move with your hand, which then moves the cursor on the screen</p> <p><u>Tablet</u>- A small, handheld computer with a touch screen</p>
Notes:	<p>Even though this practice is predominantly for practice using a mouse, your class can use a tablet. This will still allow students to practice using a touchscreen device.</p>

Week 3: Introduction to Keyboarding

Lesson overview:



Purpose:

The computer keyboard activities in this lesson are designed to help beginning computer users learn keyboard skills through hand-eye coordination by finding letters/numbers/symbols and using a finger to press the key. Teachers, please understand that your students will mainly be using the “find & peck” method of typing. That is OKAY! In this grade, our main concern is making sure that students use their left pointer finger to select keys on the left side of the keyboard and right pointer finger on the right side of the keyboard. We also want students to become familiar with finding letters/numbers/symbols on the QWERTY keyboard layout.

Lesson:

- [Keyboard L-R Coloring Sheet](#)
 - Students will identify the left/right side of the keyboard. Students will color the left side of the keyboard to coordinate with the left hand. They will color the right side of the keyboard to coordinate with the right hand.
 - The teacher will call out various letters/numbers to help students practice finding letters/numbers on a keyboard.
- Keyboarding
 - Online Option: Students can spend 15 minutes playing *Astro Bubbles* keyboarding game.
 - Unplugged Option: Using the keyboard coloring page, call out random letters and numbers and have your students find them and place their finger on them.

Lesson links/resources:

- [Keyboarding games](#)
- [Astro Bubbles Keyboard Practice](#)
- [Keyboard L-R Coloring Sheet](#) (unplugged)
- [Keyboard Coloring Page](#) (unplugged)

CS standards addressed:

Students will be able to:

- Label the letter for each key on their keyboard worksheet
- Identify the left and right side of the keyboard, and they will learn which hands to use
- Locate various letters and numbers on a keyboard

Standards:

- **CS.1A.2**—Use appropriate terminology in identifying and describing the function of common physical components of computing systems (hardware).

Time needed:

Total Time: 60 min

- Keyboard Introduction **15 min**
- Keyboarding Practice **45 min** (can be divided into 15 min sessions across three days this week)

Materials needed:

Teacher:


- Computer
- Projector/smartboard with sound
- Identifying Parts of a Computer PowerPoint presentation (for refresher)

Students:

- Markers, crayons, color pencils, etc.
- Keyboard coloring sheet
- Laptop computer, desktop computer, or tablet that is connected to the internet


Subject integrated:	ELA
Other standards addressed:	L.2.5 —Demonstrate understanding of word relationships and nuances in word meanings.
Vocabulary:	<p><u>Computer</u>: A device for working with information: data, music, videos, sounds</p> <p><u>Desktop</u>: Screen that appears if you are not browsing the internet, reading a file, or playing a game; your icons are on this screen</p> <p><u>Input</u>: To add information/data into the computer</p> <p><u>Keyboard</u>: Where all the letters, numbers, and other buttons are located; when you type on it, the symbols appear on the monitor</p> <p><u>Laptop</u>: A small, portable computer</p> <p><u>Monitor</u>: Screen that shows you what you are doing; a viewer that displays what the computer sends to it</p> <p><u>Mouse</u>: A little device you move with your hand, which then moves the cursor on the screen</p> <p><u>Output</u>: To send information/data out of the computer</p> <p><u>Tablet</u>: A small, handheld computer with a touch screen</p>
Notes:	

Week 4: We, the Digital Citizens

<p>Lesson overview:</p> 	<p>Purpose: Students will understand that being a good digital citizen means being safe and responsible online. Students will take a pledge to be a good digital citizen.</p> <p>Lesson:</p> <ul style="list-style-type: none"> ● Warm Up <ul style="list-style-type: none"> ○ Digital Citizens Coloring Book: Choose an appropriate page from this coloring book for students to color. ○ Begin the We, The Digital Citizens Slides ● Play Student Video: We, The Digital Citizens Video <ul style="list-style-type: none"> ○ Follow prompts in the lesson plan. ● Explore the Digital Citizenship Pledge ● Pause and Think <ul style="list-style-type: none"> ○ We, The Digital Citizens Hand-out <p>Any remaining time after the Common Sense Education lesson allows the students to practice keyboarding using the links from previous lessons.</p>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> ● We, The Digital Citizens: Lesson Plan ● We, The Digital Citizens Video ● We, The Digital Citizens Slides ● Digital Citizens Coloring Book ● We, The Digital Citizens Handout
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> ● Understand digital citizenship ● Understand how to be responsible online <p>Standards:</p> <ul style="list-style-type: none"> ● IC.1A.1—Compare how people live and work before and after the implementation or adoption of new computing technology. ● IC.1A.2—Work respectfully and responsibly with others online.
<p>Time needed:</p>	<p>Total Time: 50 min</p> <ul style="list-style-type: none"> ● Warm Up 5 min ● Play Student Video 2 min ● Explore 10 min ● Pause and Think 5 min ● Keyboarding of your choice ~30 min
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> ● Computer ● Projector/smartboard with sound ● We, The Digital Citizens- Lesson Plan ● We, The Digital Citizens Video ● Common Sense Account <p>Students:</p> <ul style="list-style-type: none"> ● Computer/tablet with internet access ● We, The Digital Citizens Coloring Book ● We, The Digital Citizens Hand-out ● Common Sense Account
<p>Subject integrated:</p>	<p>Social Studies</p>
<p>Other standards addressed:</p>	<p>CI.2.2—Distinguish behaviors of different individuals in the community that exhibit good citizenship.</p>

Vocabulary:	<p><u>Digital citizen</u>: Someone who uses technology safely and responsibly</p> <p><u>Pledge</u>: A promise that one makes</p>
Notes:	→Teachers will need to create FREE teacher and/or student accounts (when applicable) at https://www.commonsense.org/education/

Week 5: Device-Free Moments

<p>Lesson overview:</p> 	<p>Purpose: The students will recognize the ways in which digital devices can be distracting, identify how they feel when others are distracted by their devices, and identify ideal device-free moments for themselves and others.</p> <p>Lesson:</p> <ul style="list-style-type: none"> ● Warm Up <ul style="list-style-type: none"> ○ Follow prompts and show either Sesame Street Device-Free Dinner Video or PBS Kids "Arthur - No Internet?!" Video. ● Explore: When and where devices should be used <ul style="list-style-type: none"> ○ Talk About: safety, respect, concentration, and sleep ● Create Family Device-free Rules <ul style="list-style-type: none"> ○ Device-Free Family Handout ● Reflect on finding a good balance in our lives <ul style="list-style-type: none"> ○ Pause & Think Handout <p>Any remaining time after the Commonsense Education lesson allows the students to practice keyboarding using the links from previous lessons.</p>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> ● Device-Free Moments: Lesson plan ● Device-Free Moments Slides ● Sesame Street Device-Free Dinner Video ● PBS Kids "Arthur - No Internet?!" Video ● Device-Free Family Handout ● Pause & Think Handout
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> ● Recognize how digital devices can be distracting and create rules for device-free time <p>Standards:</p> <ul style="list-style-type: none"> ● IC.1A.1—Compare how people live and work before and after the implementation or adoption of new computing technology. ● IC.1A.2—Work respectfully and responsibly with others online.
<p>Time needed:</p>	<p>Total Time: 60 min</p> <ul style="list-style-type: none"> ● Warm Up 5 min ● Explore 10 min ● Family Device-free Rules 20 min ● Pause and Think 5 min ● Keyboarding 20 min
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> ● Computer ● Projector/smartboard with sound ● Device-Free Moments- Lesson Plan ● Device-Free Moments Slides ● Sesame Street Device-Free Dinner Video ● PBS Kids "Arthur - No Internet?!" Video ● Common Sense Account <p>Students:</p> <ul style="list-style-type: none"> ● Computer/tablet with internet access ● Device-Free Family Handout ● Pause & Think Handout ● Common Sense Account
<p>Subject integrated:</p>	<p>Social Studies</p>

Other standards addressed:	CI.2.2 —Distinguish behaviors of different individuals in the community that exhibit good citizenship.
Vocabulary:	<u>Attention</u> : Noticing someone or something as important <u>Concentration</u> : Giving your full attention to a specific activity <u>Distraction</u> : Something that keeps you from giving your full attention
Notes:	

Week 6: That's Private!

Lesson overview:



Purpose:

The students will recognize the kind of information that is private and understand that they should never give out private information online.

Lesson:

- Learn
 - Define and discuss the term "private."
 - Begin [That's Private Slides](#)
- Explore Online Forms
 - Distribute and complete page 1 of [Keep It Private Student Handout](#).
- Pause and Think
 - Complete page 2 of [Keep It Private Student Handout](#).
- Any remaining time after the Commonsense Education lesson allows the students time to practice keyboarding and using the mouse by revisiting the links from previous lessons.

Lesson links/resources:

- [That's Private! Lesson Plan](#)
- [That's Private Slides](#)
- [Keep It Private Student Handout](#)

CS standards addressed:

Students will be able to:

- Determine information that should stay private
- Know ways to keep information private

Standards:

- **IC.1A.3**—Keep login information private and log off devices appropriately.

Time needed:

Total Time: 60 min

- Learn: What is Private? **10 min**
- Explore Online Forms **15 min**
- Pause and Think **5 min**
- Keyboarding **15 min**
- Mouse Practice **15 min**

Materials needed:

Teacher:

- Computer
- Projector/smartboard with sound
- [That's Private! Lesson Plan](#)
- [That's Private Slides](#)
- [Common Sense Account](#)

Students:

- Computer/tablet with internet access
- [Keep It Private Student Handout](#)
- [Common Sense Account](#)

Subject integrated:

Social Studies

Other standards addressed:


CI.2.2—Distinguish behaviors of different individuals in the community that exhibit good citizenship.

Vocabulary:

Online: Using a computer, phone, or tablet to visit a website or app
Private: Something that you should keep to yourself

Notes:

Week 7: Digital Trails

<p>Lesson overview:</p> 	<p><u>Purpose:</u> The students will learn that the information they share online leaves a digital footprint or “trail.” Explore what information is okay to be shared online.</p> <p><u>Lesson:</u></p> <ul style="list-style-type: none"> ● Before the Lesson <ul style="list-style-type: none"> ○ Print Digital Trail Squares and follow instructions on Digital Trails-Lesson Plan. ● Watch Following the Digital Trail video. ● Play “Digital Tracks” game. <ul style="list-style-type: none"> ○ Follow the prompts for Digital Trail Squares in the lesson plan. ● Review what is okay to share. <ul style="list-style-type: none"> ○ Ask students to complete the chart on slide 10 (also located on the Digital Trails Student Handout). ● Wrap Up: Pause and think moment <ul style="list-style-type: none"> ○ Project slide 12 and ask students to summarize which information is okay to share and which is not okay to share. ○ Reflection on Digital Trails Student Handout <p>Any remaining time after the Commonsense Education lesson allows the students time to practice keyboarding using the links from previous lessons.</p>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> ● Digital Trails: Lesson plan ● Digital Trail Squares ● Following the Digital Trail: Video ● Digital Trails Slides ● Digital Trails Student Handout
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> ● Identify what information is okay to share and what information should stay private <p>Standards:</p> <ul style="list-style-type: none"> ● IC.1A.3—Keep login information private and log off of devices appropriately.
<p>Time needed:</p>	<p><u>Total Time: 60 min</u></p> <ul style="list-style-type: none"> ● Before the Lesson 5 min ● Digital Tracks 15 min ● Okay to Share? 15 min ● Wrap Up: Pause and think moment 5 min ● Keyboarding 20 min
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> ● Computer ● Projector/smartboard with sound ● Digital Trails- Lesson Plan ● Digital Trails Slides ● Following the Digital Trail: Video ● Common Sense Account <p>Students:</p> <ul style="list-style-type: none"> ● Computer/tablet with internet access ● Digital Trail Squares ● Digital Trails Student Handout ● Common Sense Account
<p>Subject integrated:</p>	<p>Social Studies</p>

Other standards addressed:	CI.2.2 —Distinguish behaviors of different individuals in the community that exhibit good citizenship.
Vocabulary:	<u>Digital footprint</u> : A record of what you do online, including the sites you visit and the things you share <u>Permanent</u> : Something that lasts forever <u>Private information</u> : Information about you that can be used to identify who you are <u>Trail</u> : A path or track
Notes:	

Week 8: Who is in Your Online Community?

Lesson overview:



Purpose:

The student will compare and contrast how they are connected to different people and places in person and on the internet, then demonstrate an understanding of how people can connect on the internet.

Lesson:

- Warm Up: What is the internet?
 - Begin [Who Is In Your Online Community Slides](#)
- Learn: How are we connected?
 - Holding a large ball or globe, discuss the prompts provided in the [Who Is In Your Online Community: Lesson Plan](#).
- Explore: Online connections
 - Discuss the [My Online Connections Handout](#) and have students work independently on the first page to list three people in their community and world.
 - Follow the prompts on the lesson plan.
- Reflect: Pause and think moment
 - Discuss prompt and have students complete page 2 on the [My Online Connections Handout](#).

Any remaining time after the Common Sense Education lesson allows the students time to practice keyboarding and using the mouse by revisiting the links from previous lessons.

Lesson links/resources:

- [Who Is in Your Online Community: Lesson plan](#)
- [Who Is in Your Online Community Slides](#)
- [My Online Connections Handout](#)

CS standards addressed:

Students will be able to:

- Compare and contrast how they are connected to different people and places around the world through the internet

Standards:

- **IC.1A.1**—Compare how people live and work before and after the implementation or adoption of new computing technology.
- **IC.1A.2**—Work respectfully and responsibly with others online.

Time needed:

Total Time: 60 min

- Warm Up: What is the internet? **5 min**
- Learn: How are we connected? **10 min**
- Explore: Online connections **10 min**
- Reflect: Pause and think moment" **5 min**
- Keyboarding practice twice more during this week using the links provided in previous lessons **15 min**

Materials needed:

Teacher:

- Computer
- Projector/smartboard with sound
- Large ball or globe
- [Who Is in Your Online Community Lesson Plan](#)
- [Who Is in Your Online Community Slides](#)
- [Common Sense Account](#)

Students:

- Computer/tablet with internet access
- [My Online Connections Handout](#)
- [Common Sense Account](#)

Subject integrated:

Social Studies

Other standards addressed:	<ul style="list-style-type: none">• CI.2.2.1—Demonstrate knowledge of how to be a good citizen in the local community.• CR.2.1.1—Illustrate the role of unity and diversity within the community.
Vocabulary:	<p><u>Community</u>: People who share a common neighborhood, background, or interests</p> <p><u>Internet</u>: A worldwide network that connects people using computers, phones, or other devices</p>
Notes:	

Week 9: Let's Give Credit

Lesson overview:



Purpose:

The student will explain how giving credit signals respect for people's work and learn how to give credit in their schoolwork for content they use from the internet.

Lesson:

- Warm Up: That's Mine!
 - Begin [Let's Give Credit Slides](#)
 - Follow prompts on [Let's Give Credit Lesson Plan](#)
- Explore: How do we give credit?
 - Follow slides and prompts
- Create: Research report
 - [Let's Give Credit Student Report](#)
- Reflect: Pause and think moment
 - [Pause and Think Handout](#)

Any remaining time after the Common Sense Education lesson allows the students time to practice keyboarding and mouse techniques using the links from previous lessons.

Lesson links/resources:

- [Let's Give Credit Lesson Plan](#)
- [Let's Give Credit Slides](#)
- [Let's Give Credit Teacher Report](#)
- [Let's Give Credit Student Report](#)
- [Pause and Think Handout](#)

CS standards addressed:

Students will be able to:

- Show how to give credit to other people's work

Standards:

- **IC.1A.2**—Work respectfully and responsibly with others online.

Time needed:

Total Time: 60 min

- Warm Up: That's Mine! **5 min**
- Explore: How Do We Give Credit? **10 min**
- Create Research Report **10 min**
- Reflect: Pause and think moment **5 min**
- Keyboarding **30 min** (can be divided into 15 min sessions across two days this week)

Materials needed:

Teacher:

- Computer
- Projector/smartboard with sound
- [Let's Give Credit Lesson Plan](#)
- [Let's Give Credit Slides](#)
- [Let's Give Credit Teacher Report](#)
- [Common Sense Account](#)

Students:

- Computer/tablet with internet access
- [Let's Give Credit Student Report](#)
- [Pause and Think Handout](#)
- [Common Sense Account](#)

Subject integrated:

Social Studies

Other standards addressed:


CI.2.2—Distinguish behaviors of different individuals in the community that exhibit good citizenship.

Vocabulary:

Credit: Giving recognition to a person that created something
Respect: Showing that you appreciate someone

Notes:

Week 10: Code.org, Course C, Lesson 1—Putting a STOP to Online Meanness

<p>Lesson overview:</p> 	<p>Purpose: Common Sense Education created this lesson to teach students about meanness and what they should do if they encounter it online.</p> <p>Lesson:</p> <ul style="list-style-type: none"> ● Learn: What is meanness? <ul style="list-style-type: none"> ○ Begin Putting a STOP to ONLINE Meanness Slides ○ Review key vocabulary ● Perform: Online vs. face-to-face <ul style="list-style-type: none"> ○ Follow prompts ● Explore: Stop meanness <ul style="list-style-type: none"> ○ STOP Online Meanness Student Handout ○ Follow prompts ● Wrap Up: Pause and think <ul style="list-style-type: none"> ○ Follow prompts and complete handout
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> ● Code.org, Lesson 1: Putting a STOP to Online Meanness Lesson Plan ● Putting a STOP to Online Meanness Slides ● STOP Online Meanness Student Handout
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> ● Identify ways to respond to mean words online, using S-T-O-P ● Understand what online meanness can look like and how it can make people feel <p>Standards:</p> <ul style="list-style-type: none"> ● IC.1A.2—Work respectfully and responsibly with others online.
<p>Time needed:</p>	<p>Total Time: 60 min</p> <ul style="list-style-type: none"> ● Learn: What is meanness? 10 min ● Perform: Online vs. face-to-face 5 min ● Explore: Stop meanness 15 min ● Wrap Up: Pause and think 5 min ● Keyboarding games 25 min
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> ● Computer ● Projector/smartboard with sound ● Code.org account ● Code.org Lesson 1: Putting a STOP to Online Meanness Lesson Plan ● Putting a STOP to Online Meanness Slides <p>Students:</p> <ul style="list-style-type: none"> ● Computer/Tablet with internet access ● Code.org account ● STOP Online Meanness Student Handout
<p>Subject integrated:</p>	<p>Social Studies</p>
<p>Other standards addressed:</p>	<ul style="list-style-type: none"> ● CI.2.2.1—Demonstrate knowledge of how to be a good citizen in the local community. ● CR.2.3.1—Explain the role of cooperation and compromise within the community.
<p>Vocabulary:</p>	<p><u>Meanness</u>: The state or quality of being mean <u>Respect</u>: To consider worthy of high regard</p>

Notes:

→ Teachers will need to create FREE teacher and/or student accounts
https://studio.code.org/users/sign_in

Week 11: Code.org, Course C, Lesson 2—My Robotic Friends Jr.

Lesson overview:



Purpose:

This unplugged lesson brings the class together as a team with a simple task of getting a "robot" to stack cups in a specific design. This activity lays the groundwork for the programming that students will do throughout the course as they learn the importance of defining a clearly communicated algorithm.

Lesson:

- Before the Lesson
 - Watch [My Robotic Friends-Unplugged Video \(Download\)](#)
- Warm Up
 - Talking to robots: Watch one of the following videos to demonstrate what robots can do.
 - [Asimo by Honda](#) (3:58)
 - [Egg Drawing Robot](#) (3:15)
 - [Dancing Lego Robot](#) (1:35)
 - Follow prompts
- Activity
 - My Robotic Friends
 - Follow prompts
- Wrap Up
 - Reflection
 - Draw your own stack of cups that you would like to see a robot build.
 - Can you create a program for that cup stack?

Lesson links/resources:

- [Code.org, Lesson 2: My Robotic Friends Jr.](#)
- [My Robotic Friends](#): Cup Spacing
- [My Robotic Friends](#): Symbol Key
- [My Robotic Friends](#): Unplugged Video ([Download](#))
- [My Robotic Friends](#): Cup Stacking Ideas
- [My Robotic Friends](#): Paper Trapezoid Template

CS standards addressed:

Students will be able to:

- Attend to precision when creating instructions
- Identify and address bugs or errors in sequenced instructions

Standards:

- **AP.1A.2**—Model the way programs store and manipulate data by using numbers or other symbols to represent information.

Time needed:

Total Time: 60 min

- Warm Up: Talking to robots **5 min**
- Activity: My Robotic Friends **30 min**
- Wrap Up **10 min**
- Keyboarding **15 min**

Materials needed:

Teacher:

- Computer
- Projector/smartboard with sound
- [Code.org account](#)
- [My Robotic Friends](#)- Unplugged Video ([Download](#))

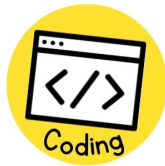
Students:

- Computer/tablet with internet access
- [My Robotic Friends](#): Cup spacing
- [My Robotic Friends](#): Symbol key
- [My Robotic Friends](#): Cup stacking ideas
- [My Robotic Friends](#): Paper trapezoid template

Subject integrated:	ELA Math
Other standards addressed:	ELA <ul style="list-style-type: none">● SL.2.1—Participate in collaborative conversations with diverse partners about Grade 2 topics and texts with peers and adults in small and larger groups. Math <ul style="list-style-type: none">● 2.MD.9—Generate measurement data by measuring lengths of several objects to the nearest whole unit, or by making repeated measurements of the same object. Show the measurements by making a line plot, where the horizontal scale is marked off in whole-number units.
Vocabulary:	<u>Algorithm</u> : A list of steps to finish a task <u>Bug</u> : Part of a program that does not work correctly <u>Debugging</u> : Finding and fixing problems in an algorithm or program <u>Program</u> : An algorithm that has been coded into something that can be run by a machine
Notes:	

Week 12: Code.org, Course C, Lesson 3—Programming With Angry Birds

Lesson overview:



Purpose:

In this lesson, students will develop programming and debugging skills on a computer platform. The block-based format of these puzzles helps students learn about sequence and concepts without having to worry about perfecting syntax.

Lesson:

- Warm Up: Introduction
 - Review My Robotic Friends activity from Week 11
 - Follow prompts
- Bridging Activity: Programming
 - Display [Maze Bridging Page](#) for students to see
 - Follow prompts
- Previewing Online: Puzzles as a class
 - Pull up puzzle 5 to do in front of the class
 - Follow prompts
- Main Activity: Programming with Angry Birds
 - Students will watch [Maze Intro-Programming with Blocks](#)
 - Students will only complete all levels of Programming with Angry Birds
- Wrap Up: Reflection
 - What was today's lesson about?
 - How did you feel during today's lesson?
 - Draw an activity to do that you struggled with the first time. Draw or describe how you got better.

Lesson links/resources:

- [Code.org, Lesson 3: Programming with Angry Birds](#)
- [Maze Intro: Programming with Blocks](#) video
- [Maze Bridging Page](#)
- [Unplugged Maze Blocks](#): Manipulatives

CS standards addressed:

Students will be able to:

- Identify and locate bugs in a program
- Translate movements into a series of commands

Standards:

- **AP.1A.2**—Model the way programs store and manipulate data by using numbers or other symbols to represent information.
- **AP.1A.4**—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.

Time needed:

Total Time: 55 min

- Warm Up: Introduction **5 min**
- Bridging Activity: Programming **10 min**
- Previewing Online: Puzzles as a class **5 min**
- Main Activity: Programming with Angry Birds **30 min**
- Wrap Up **5 min**

Materials needed:

Teacher:


- Computer
- Projector/smartboard with sound
- [Maze Bridging Page](#)
- [Maze Intro: Programming with Blocks](#): Video
- [Code.org](#) account

Students:

- Computer/tablet with internet access
- [Unplugged Maze Blocks](#): Manipulatives

	<ul style="list-style-type: none"> • Code.org account
Subject integrated:	<p>ELA Math</p>
Other standards addressed:	<p>ELA</p> <ul style="list-style-type: none"> • L.2.1—Demonstrate command of the conventions of standard English grammar and usage when writing (printing, cursive, or keyboarding) or speaking. • L.2.2—Demonstrate command of the conventions of standard English capitalization, punctuation, and spelling when writing. <p>Math</p> <ul style="list-style-type: none"> • 2.OA.2—Fluently add and subtract within 20 using mental strategies. By the end of Grade 2, know from memory all sums of two one-digit numbers. • 2.NBT.4—Compare two three-digit numbers based on meanings of the hundreds, tens, and ones digits, using $>$, $=$, and $<$ symbols to record the results of comparisons.
Vocabulary:	<p><u>Algorithm</u>: A list of steps to finish a task <u>Bug</u>: Part of a program that does not work correctly <u>Debugging</u>: Finding and fixing problems in an algorithm or program <u>Sequencing</u>: Putting commands in correct order so computers can read the commands</p>
Notes:	

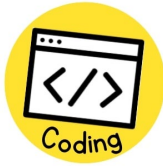
Week 13: Code.org, Course C, Lesson 3 (Extension): Bugs, Bugs, and More Bugs

<p>Lesson overview:</p> 	<p>Purpose: Whether fixing code, correcting a math problem or proofreading writing, debugging is an important part of life. Learning how to spot errors and knowing how to correct them will help prevent bugs, bugs, and more bugs!</p> <p>Lesson:</p> <ul style="list-style-type: none"> ● Note <ul style="list-style-type: none"> ○ Students completed Programming with Angry Bird, Levels 1-11 in Week 12. ● Prompt <ul style="list-style-type: none"> ○ Did you feel frustrated while debugging your code? Explain why or why not debugging was frustrating to you. ● Video <ul style="list-style-type: none"> ○ Watch the How to Debug video. ● Bugs, Bugs, and More Bugs: Student Handout <ul style="list-style-type: none"> ○ Debugging code ○ Debugging Math Problems ○ Debugging Sentences ● Wrap Up <ul style="list-style-type: none"> ○ Have students pair up to discuss the following prompt: <ul style="list-style-type: none"> ■ Today you had practice debugging code, math problems, and sentences. How do you use debugging in other areas of your life?
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> ● Bugs, Bugs, and More Bugs: Lesson plan ● Bugs, Bugs, and More Bugs: Student handout
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> ● Identify and locate bugs in a program ● Translate movements into a series of commands <p>Standards:</p> <ul style="list-style-type: none"> ● AP.1A.2—Model the way programs store and manipulate data by using numbers or other symbols to represent information. ● AP.1A.4—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.
<p>Time needed:</p>	<p>Total Time: 48 min</p> <ul style="list-style-type: none"> ● Discuss Prompt 5 min ● Video 3 min ● Student Handout 30 min ● Wrap Up 10 min
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> ● Computer ● Projector/smartboard with sound ● Bugs, Bugs & More Bugs Lesson Plan ● Code.org account <p>Students:</p> <ul style="list-style-type: none"> ● Computer/tablet with internet access ● Paper ● Bugs, Bugs, and More Bugs Student Handout
<p>Subject integrated:</p>	<p>ELA Math</p>

Other standards addressed:	<p>ELA</p> <ul style="list-style-type: none">• L.2.1—Demonstrate command of the conventions of standard English grammar and usage when writing (printing, cursive, or keyboarding) or speaking.• L.2.2—Demonstrate command of the conventions of standard English capitalization, punctuation, and spelling when writing. <p>Math</p> <ul style="list-style-type: none">• 2.OA.2—Fluently add and subtract within 20 using mental strategies. By the end of Grade 2, know from memory all sums of two one-digit numbers.• 2.NBT.4—Compare two three-digit numbers based on meanings of the hundreds, tens, and ones digits, using $>$, $=$, and $<$ symbols to record the results of comparisons.
Vocabulary:	<p><u>Algorithm</u>: A list of steps to finish a task</p> <p><u>Bug</u>: Part of a program that does not work correctly</p> <p><u>Debugging</u>: Finding and fixing problems in an algorithm or program</p> <p><u>Sequencing</u>: Putting commands in correct order so computers can read the commands</p>
Notes:	

Week 14: Code.org, Lesson 4—Debugging in Maze

Lesson overview:



Purpose:

Students in your class might become frustrated with this lesson because of the essence of debugging. Debugging is a concept that is very important to computer programming. Computer scientists have to get really good at facing the bugs in their own programs. Debugging forces the students to recognize problems and overcome them while building critical thinking and problem-solving skills.

Lesson:

- Warm Up
 - Introduction: Debugging
 - Follow prompts
- Main Activity
 - Debugging in Maze
 - Students will watch the [Debugging with Step Button](#) video and work through the levels in this lesson.
- Wrap Up: Reflection
 - What was today's lesson about?
 - How did you feel during today's lesson?
 - What kind of bugs did you find today?
 - Draw a bug you encountered in one of the puzzles today. What did you do to debug the program?

Lesson links/resources:

- [Code.org, Lesson 4 Debugging in Maze Lesson Plan](#)
- [Pair Programming Video](#)
- [Debugging with Step Button](#)
- [Main Activity Notes](#)
- [Debugging Guide](#)

CS standards addressed:

Students will be able to:

- Modify an existing program to solve errors
- Predict where a program will fail
- Reflect on the debugging process in an age-appropriate way

Standards:

- **AP.1A.2**—Model the way programs store and manipulate data by using numbers or other symbols to represent information.
- **AP.1A.4**—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.

Time needed:

Total Time: 60 min

- Warm Up: Introduction to debugging **15 min**
- Debugging in Maze **30 min**
- Wrap Up: Reflection (prompts provided) **10 min**
- Keyboarding **5 min**

Materials needed:

Teacher:

- Computer
- Projector/smartboard with sound
- [Code.org, Lesson 4: Debugging in Maze Lesson Plan](#)
- [Pair programming video](#)
- [Debugging with step button](#)
- [Main activity notes](#)
- [Code.org](#) account

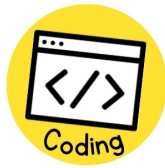
Students:

- Computer/tablet with internet access
- [Debugging guide](#)

	<ul style="list-style-type: none">• Code.org account
Subject integrated:	ELA
Other standards addressed:	SL.2.1 —Participate in collaborative conversations with diverse partners about Grade 2 topics and texts with peers and adults in small and larger groups.
Vocabulary:	<u>Bug</u> : Part of a program that does not work correctly <u>Debugging</u> : Finding and fixing problems in an algorithm or program
Notes:	

Week 15: Code.org, Course C, Lesson 5—Collecting Treasure With Laurel

Lesson overview:



Purpose:

In this lesson, students will be practicing their programming skills using a new character, Laurel the Adventurer. When someone starts programming, they piece together instructions in a specific order using something that a machine can read. Through the use of programming, students will develop an understanding of how a computer navigates instructions and order. Using a new character with a different puzzle objective will help students widen their scope of experience with sequencing and algorithms in programming.

Lesson:

- Warm Up: Introduction to collecting
 - Bridging activity: Preview online puzzles
 - Pull a puzzle from the corresponding online stage. We recommend puzzle 7. Have students discuss a pattern that they think will get Laurel the Adventurer to collect all the treasure. Ask the students to share. See how many other students had the same answer!
- Main Activity
 - Collecting Treasure with Laurel
- Wrap Up:
 - What was today's lesson about?
 - How did you feel during today's lesson?
 - Draw a maze that you might solve with the blocks you used today.

Lesson links/resources:

- [Code.org, Lesson 5: Collecting Treasure with Laurel Lesson Plan](#)
- [Unplugged Blocks: Manipulatives](#)

CS standards addressed:

Students will be able to:

- Develop problem solving and critical thinking skills by reviewing debugging practices
- Order movement commands as sequential steps in a program
- Represent an algorithm as a computer program

Standards:

- **AP.1A.2**—Model the way programs store and manipulate data by using numbers or other symbols to represent information.
- **AP.1A.4**—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.

Time needed:

Total Time: 60 min

- Warm Up: Introduction to collecting **5 min**
- Preview Online Puzzles **10 min**
- Collecting Treasure with Laurel **30 min**
- Wrap Up **5 min**
- Keyboarding **10 min**

Materials needed:

Teacher:

- Computer
- Projector/smartboard with sound
- [Code.org, Lesson 5: Collecting Treasure with Laurel Lesson Plan](#)
- [Code.org](#) account

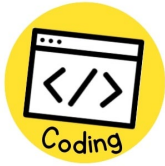
Students:

- Computer/tablet with internet access
- [Unplugged Blocks: Manipulatives](#)
- [Code.org](#) account

Subject integrated:	Social Studies
Other standards addressed:	<ul style="list-style-type: none">• G.2.1.1—Differentiate between different types of maps.• G.2.2.3—Investigate physical features of the local region.
Vocabulary:	<p><u>Algorithm</u>: A list of steps to finish a task</p> <p><u>Program</u>: An algorithm that has been coded into something that can be run by a machine</p> <p><u>Programming</u>: The art of creating a program</p>
Notes:	

Week 16: Code.org, Course C, Lesson 6—Creating Art With Code

Lesson overview:



Purpose:

Building off of the students' previous experience with sequencing, this lesson will work to inspire more creativity with coding. The purpose of this lesson is to solidify knowledge on sequencing by introducing new blocks and goals. In this case, students learn more about pixels and angles using the new blocks, while still practicing their sequencing skills. Also, students will be able to visualize new goals such as coding the artist to draw a square.

Lesson:

- Warm Up: Introduction to angles
 - [Artist Introduction](#): Student video
 - [Turns and Angles](#): Student video
- Main Activity
 - Creating art with code
- Wrap Up
 - What was today's lesson about?
 - How did you feel during today's lesson?
 - What are the interior angles that make up a square? What about a triangle?
 - Sketch a simple shape on your paper and imagine the code used to draw it. Can you write that code out next to the shape?

Lesson links/resources:

- [Code.org, Lesson 6: Creating Art with Code](#)
- [Artist Introduction](#): Student video
- [Turns & Angles](#): Student video
- [Turns & Angles](#): Student handout

CS standards addressed:

Students will be able to:

- Break complex shapes into simple parts
- Create a program to complete an image using sequential steps

Standards:

- **AP.1A.2**—Model the way programs store and manipulate data by using numbers or other symbols to represent information.
- **AP.1A.4**—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.
- **AP.1A.7**—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.

Time needed:

Total Time: 60 min

- Warm Up: Introduction to angles **10 min**
- Activity: Creating art with code **30 min**
- Wrap Up: Reflection (prompts provided) **10 min**
- Keyboarding **10 min**

Materials needed:

Teacher:

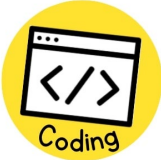
- Computer
- Projector/smartboard with sound
- [Code.org, Lesson 6: Creating Art with Code](#)
- [Code.org](#) account

Students:

- Computer/tablet with internet access
- [Artist Introduction](#): Student video
- [Turns & Angles](#): Student video
- [Turns & Angles](#): Student handout
- [Code.org](#) account

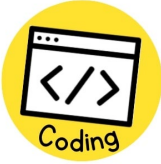

Subject integrated:	Math
Other standards addressed:	2.G.1 —Recognize and draw shapes having specified attributes, such as a given number of angles or a given number of equal faces. Identify triangles, quadrilaterals, pentagons, hexagons, and cubes.
Vocabulary:	<u>Trapezoid</u> : A quadrilateral with at least one pair of parallel sides
Notes:	

Week 17: Code.org, Course C, Lesson 6 (Cross-Curricular)—Shapes and Landscapes

<p>Lesson overview:</p> 	<p>Purpose: Students are acting as an engineer at the USA Dam Construction Company. A local community is having issues with flooding. Every time it floods, soil washes away and makes the roads unsafe. Students have been asked to design a dam to prevent future flooding. Using code, you will create a blueprint to show the local town council how your dam will look. Building off of the students' previous experience with sequencing, this lesson will work to inspire more creativity with coding. The purpose of this lesson is to solidify knowledge on sequencing by introducing new blocks and goals. In this case, students learn more about pixels and angles using the new blocks while still practicing their sequencing skills. Also, students will be able to visualize new goals such as coding the artist to draw a square.</p> <p>Lesson:</p> <ul style="list-style-type: none"> ● Warm Up <ul style="list-style-type: none"> ○ Review students on dams and other water structures that may shape the land and change the direction of water. ○ Artist Introduction - Student Video (1.5 minutes long) ○ Turns and Angles - Student Video (2 minutes long) ● Main Activity: Creating art with code <ul style="list-style-type: none"> ○ Students will pair program to complete Creating Art with Code Levels 2-7. The Turns & Angles student handout is available to assist students as they learn to code the various shapes. ○ Each student will complete the Shapes and Landscapes Student Handout. Students will use Creating Art with Code, Level 2 to complete their dam blueprint. ○ To wrap up, have students complete the "L - Write what you learned" box on the KWL Chart located on the Shapes and Landscapes Student Handout. ● Wrap Up: Optional extension <ul style="list-style-type: none"> ○ Shapes & Landscapes Student Handout: Besides building a dam, what other solutions could you use to slow or prevent wind or water from changing the shape of the land? <ul style="list-style-type: none"> ● Examples of solutions could include different designs of dikes and windbreaks to hold back wind and water, and different designs for using shrubs, grass, and trees to hold back the land.
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> ● Code.org, Lesson 6: Shapes and Landscapes ● Artist Introduction: Student Video (1.5 minutes long) ● Turns & Angles: Student Video (2 minutes long) ● Turns & Angles: Student Handout ● Shapes & Landscapes Student Handout
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> ● Break complex shapes into simple parts ● Create a program to complete an image using sequential steps <p>Standards:</p> <ul style="list-style-type: none"> ● AP.1A.2—Model the way programs store and manipulate data by using numbers or other symbols to represent information. ● AP.1A.4—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.

	<ul style="list-style-type: none"> ● AP.1A.7—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.
Time needed:	<p>Total Time: 60 min</p> <ul style="list-style-type: none"> ● Warm Up 10 min ● Main Activity: Creating art with code 30 min ● Wrap Up: Reflection 10 min
Materials needed:	<p>Teacher:</p> <ul style="list-style-type: none"> ● Computer ● Projector/smartboard with sound ● Code.org, Lesson 6: Shapes and Landscapes ● Artist Introduction: Student video ● Turns & Angles: Student video ● Code.org account <p>Students:</p> <ul style="list-style-type: none"> ● Computer/tablet with internet access ● Turns & Angles: Student handout ● Shapes & Landscapes Student Handout ● Code.org account
Subject integrated:	Math Science
Other standards addressed:	<p>Math</p> <ul style="list-style-type: none"> ● 2.MD.3—Estimate lengths using units of inches, feet, centimeters, and meters. <p>Science</p> <ul style="list-style-type: none"> ● E.2.10.1—Using informational text, other media, and first-hand observations to investigate, analyze and compare the properties of Earth materials (including rocks, soils, sand, and water). ● E.2.10.4—Using informational text, other media, and first-hand observations to investigate and communicate the process and consequences of soil erosion. ● E.2.10.5—With teacher guidance, investigate possible solutions to prevent or repair soil erosion.
Vocabulary:	<u>Decompose</u> : To break down or break apart
Notes:	

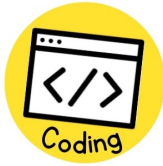
Week 18: Code.org, Course C, Lesson 7: My Loopy Robotic Friends Jr.

<p>Lesson overview:</p>  <p>Coding</p>  <p>Unplugged</p>	<p>Purpose: This lesson serves as a reintroduction to loops, using the now familiar set of "robot" programming instructions. Students will develop critical thinking skills by looking for patterns of repetition in the movements of classmates and determining how to simplify those repeated patterns using loops.</p> <p>Lesson:</p> <ul style="list-style-type: none">• Warm Up: My Robotic Friends review<ul style="list-style-type: none">◦ Display My Robotic Friends: Symbol key◦ Pull a puzzle from My Robotic Friends: Cup stacking ideas◦ Follow prompts in Code.org- Lesson 7: My Loopy Robotic Friends, Jr. Lesson Plan• Introduce My Loopy Friend by showing Unplugged Activity: My Loopy Robotic Friends video• Activity: My Loopy Robotic Friends<ul style="list-style-type: none">◦ Follow prompts in lesson plan◦ Think, pair, share, model◦ Looping your robots: Follow prompts• Wrap Up: Reflection<ul style="list-style-type: none">◦ Have the students write or draw something in their journal, or on a piece of paper, that will remind them what loops are. This can come from a prompt like:<ul style="list-style-type: none">■ What does "repeat" mean to you?■ Draw a picture of you repeating something.
<p>Lesson links/resources:</p>	<ul style="list-style-type: none">• Code.org, Lesson 7: My Loopy Robotic Friends Jr. Lesson Plan• My Robotic Friends: Symbol Key• My Robotic Friends: Cup Stacking Ideas• My Robotic Friends: Cup Spacing• My Robotic Friends: Paper Trapezoid Template• Unplugged Activity: My Loopy Robotic Friends video• Feeling Faces Emotion Image- Resource
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none">• Identify repeated patterns in code that could be replaced with a loop• Write instructions that use loops to repeat patterns <p>Standards:</p> <ul style="list-style-type: none">• AP.1A.2—Model the way programs store and manipulate data by using numbers or other symbols to represent information.• AP.1A.3—Develop programs with sequences and simple loops to express ideas or address a problem.• AP.1A.4—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.• AP.1A.7—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.
<p>Time needed:</p>	<p>Total: 60 min</p> <ul style="list-style-type: none">• Warm Up: My Robotic Friends review 10 min• Activity: My Loopy Robotic Friends 30 min• Wrap Up: Reflection (prompts provided) 5 min• Keyboarding 15 min
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none">• Computer• Projector/smartboard with sound

	<ul style="list-style-type: none"> • Code.org, Lesson 7: My Loopy Robotic Friends Jr. Lesson Plan • My Robotic Friends: Symbol key • My Robotic Friends: Cup stacking ideas • Code.org account <p>Students:</p> <ul style="list-style-type: none"> • Journal • Stacking cups (20 per each group of 4 students) • My Robotic Friends: Symbol key • My Robotic Friends: Cup stacking ideas • My Robotic Friends: Cup spacing • My Robotic Friends: Paper trapezoid template
Subject integrated:	ELA Math
Other standards addressed:	<p>ELA</p> <ul style="list-style-type: none"> • W.2.2—Write informative/explanatory texts in which they introduce a topic, use facts and definitions to develop points, and provide a concluding statement or section. • W.2.5—With guidance and support from adults and peers, focus on a topic and strengthen writing as needed by revising and editing. <p>Math</p> <ul style="list-style-type: none"> • 2.OA.2—Fluently add and subtract within 20 using mental strategies. By the end of Grade 2, know from memory all sums of two one-digit numbers.
Vocabulary:	<p><u>Loop</u>: The action of doing something over and over again</p> <p><u>Repeat</u>: To do something again</p>
Notes:	

Week 19: Code.org, Course C, Lesson 8: Loops With Rey and BB-8

Lesson overview:



Purpose:

In this lesson, students will be learning more about loops and how to implement them in Blockly code. Using loops is an important skill in programming because manually repeating commands is tedious and inefficient. With the Code.org puzzles, students will learn to add instructions to existing loops, gather repeated code into loops, and recognize patterns that need to be looped. It should be noted that students will face puzzles with many different solutions. This will open up discussions on the various ways to solve puzzles with advantages and disadvantages to each approach.

Lesson:

- Warm Up: Introduction—My Loopy Robotic Friends
 - Follow prompts in [Code.org, Lesson 8: Loops with Rey and BB-8 Lesson Plan](#)
- Bridging Activity
 - Choose 1—paper blocks or previewing online puzzles
- Loops with BB-8
- Wrap Up: Reflection
 - What was today's lesson about?
 - How did you feel during today's lesson?
 - How did loops make your program easier to write?
 - Think of something that repeats over and over again. What might the program for that look like?

Lesson links/resources:

- [Code.org, Lesson 8: Loops with Rey and BB-8 Lesson Plan](#)
- [Unplugged Blocks \(Courses C-F\): Manipulatives](#)
- [Pair Programming: Video](#)

CS standards addressed:

Students will be able to:

- Break down a long sequence of instructions into the largest repeatable sequence
- Employ a combination of sequential and looped commands to reach the end of a maze
- Identify the benefits of using a loop structure instead of manual repetition

Standards:

- **AP.1A.2**—Model the way programs store and manipulate data by using numbers or other symbols to represent information.
- **AP.1A.3**—Develop programs with sequences and simple loops to express ideas or address a problem.
- **AP.1A.4**—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.
- **AP.1A.7**—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.

Time needed:

Total Time: 60 min

- Warm Up **5 min**
- Bridging Activity **10 min**
- Activity: Loops with BB-8 **30 min**
- Wrap Up: Reflection **5 min**
- Keyboarding **10 min**

Materials needed:

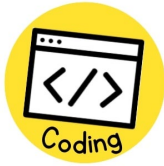
Teacher:

- Computer
- Projector/smartboard with sound

	<ul style="list-style-type: none"> • Code.org, Lesson 8: Loops with Rey and BB-8 Lesson Plan • Code.org account <p>Students:</p> <ul style="list-style-type: none"> • Computer/tablet with internet access • Unplugged Blocks (Courses C-F): Manipulatives • Pair Programming: Video • Code.org account
Subject integrated:	<p>ELA Math Science</p>
Other standards addressed:	<p>ELA</p> <ul style="list-style-type: none"> • W.2.1—Write opinion pieces in which they introduce the topic or book they are writing about, state an opinion, supply reasons that support the opinion, use linking words (e.g., because, and, also) to connect opinion and reasons, and provide a concluding statement or section. <p>Math</p> <ul style="list-style-type: none"> • 2.G.A.1—Recognize and draw shapes having specified attributes, such as a given number of angles or a given number of equal faces.1 Identify triangles, quadrilaterals, pentagons, hexagons, and cubes.
Vocabulary:	<p><u>Sketch</u>: To draw <u>Attributes</u>: Characteristics</p>
Notes:	

Week 20: Code.org, Course C, Lesson 9: Harvesting Crops With Loops

Lesson overview:



Purpose:

In this lesson, students will use loops to repeat actions like harvesting pumpkins. New patterns will emerge, and students will use creativity and logical thinking to determine what code needs to be repeated and how many times.

Lesson:

- Warm Up: Introduction
 - Follow prompts in [Code.org, Lesson 9: Harvesting Crops with Loops Lesson Plan](#)
- Activity
 - Harvesting Crops with Loops
- Wrap Up
 - What was today's lesson about?
 - How did you feel during today's lesson?
 - Give two examples of when you used loops in your code.
 - What else could a farmer harvest? Draw the code block that you would need to harvest that item.

Lesson links/resources:

[Code.org, Lesson 9: Harvesting Crops with Loops Lesson Plan](#)

CS standards addressed:

Students will be able to:

- Employ a combination of sequential and looped commands to move and perform actions
- Identify when a loop can be used to simplify a repetitive action
- Write a program for a given task which loops a single command

Standards:

- **AP.1A.2**—Model the way programs store and manipulate data by using numbers or other symbols to represent information.
- **AP.1A.3**—Develop programs with sequences and simple loops to express ideas or address a problem.
- **AP.1A.4**—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.
- **AP.1A.7**—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.

Time needed:

Total Time: 60 min

- Warm Up: Introduction **5 min**
- Harvesting Crops with Loops **30 min**
- Wrap Up: Reflection (prompts provided) **10 min**
- Keyboarding **15 min**

Materials needed:

Teacher:

- Computer
- Projector/smartboard with sound
- [Code.org, Lesson 9: Harvesting Crops with Loops Lesson Plan](#)
- [Code.org](#) account

Students:

- Computer/tablet with internet access
- [Code.org](#) account

Subject integrated:

Social Studies

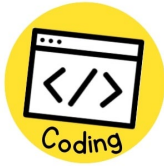
Other standards addressed:

E.2.1—Explain how individual wants and needs impact the production of goods and services.

Vocabulary:	<u>Loop</u> : The action of doing something over and over again <u>Repeat</u> : To do something again
Notes:	

Week 21: Code.org, Course C, Lesson 9—Harvester's Farm

Lesson overview:



Purpose:

Harvester needs help describing and classifying items on her farm. Students will use the provided graphic organizers to classify and describe these items. The graphic organizers will serve as a pre-writing component as students then write an informative/explanatory text to inform readers about what the Harvester does on her farm.

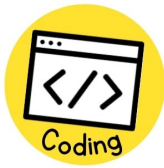
Lesson:

- Main Activity
 - Students will pair program to complete [Harvesting Crops with Loops, Levels 1-12](#). In addition to helping the driver proofread code, the navigator will also add the total number of crops harvested at each level. Students should complete this addition using **mental** arithmetic strategies. Then record their answer on slide 1 of the [Harvester's Farm Graphic Organizers](#) file.
 - The role of driver and navigator will alternate at each level.
 - Students will open the Google Slide file titled [Harvester's Farm Graphic Organizers](#). Each student will want to "Make a Copy" of this file so they can add their own answers to the slides.
 - On the [Harvester's Farm Graphic Organizers](#) file, students will follow the instructions provided on each slide in the "speaker notes" section.
 - **Bubble Map instructions:** Help Harvester describe the crop located in the center bubble/circle. You will write 1-2 word descriptions in each of the surrounding bubbles/circles. Remember that your descriptions could include characteristics such as color, texture, shape, etc.
 - **Tree Map instructions:** Harvester has many different things on her farm. Classify each item by its observable properties as either an animal, plant or tool. Then use your mouse to drag the picture to the correct location on the tree map.
 - Each student will write an informative/explanatory text about Harvester's farm. This can be done with paper and pencil OR digital tools such as Google docs. Their writing should include the following:
 - Introduction of the topic: Describe what the Harvester does on her farm.
 - Use facts and definitions to develop points
 - Provide a concluding statement
- Wrap Up
 - Group 3-4 students together and have them read their informative/explanatory text to each other. Students will then provide suggestions for revisions and edits which could focus and strengthen each other's writing.

Lesson links/resources:	<ul style="list-style-type: none"> • Harvester's Farm Lesson Plan • Harvesting Crops with Loops, Levels 1-12 • Harvester's Farm Graphic Organizers
CS standards addressed:	<p>Students will be able to:</p> <ul style="list-style-type: none"> • Employ a combination of sequential and looped commands to move and perform actions • Identify when a loop can be used to simplify a repetitive action • Write a program for a given task which loops a single command <p>Standards:</p> <ul style="list-style-type: none"> • AP.1A.5—Develop plans that describe a program's sequence of events, goals, and expected outcomes. • AP.1A.8—Using correct terminology, describe steps taken and choices made during the iterative process of program development.
Time needed:	<p>Total Time: 55 min</p> <ul style="list-style-type: none"> • Main Activity 30 min • Wrap Up 10 min • Keyboarding 15 min
Materials needed:	<p>Teacher:</p> <ul style="list-style-type: none"> • Computer • Projector/smartboard with sound • Harvester's Farm Lesson Plan • Code.org account <p>Students:</p> <ul style="list-style-type: none"> • Harvester's Farm Graphic Organizers (Bubble Maps and Tree Map on Google Slides) • Paper and pencil (optional)
Subject integrated:	ELA, Math, Science
Other standards addressed:	<p>ELA</p> <ul style="list-style-type: none"> • W.2.2—Write informative/explanatory texts in which they introduce a topic, use facts and definitions to develop points, and provide a concluding statement or section. • W.2.5—With guidance and support from adults and peers, focus on a topic and strengthen writing as needed by revising and editing. <p>Math</p> <ul style="list-style-type: none"> • 2.OA.2—Fluently add and subtract within 20 using mental strategies. By the end of Grade 2, know from memory all sums of two-digit numbers. <p>Science</p> <ul style="list-style-type: none"> • P.2.6.1—Conduct a structured investigation to collect, represent, and analyze data from observations and measurements to demonstrate the effects of pushes and pulls with different strengths and directions. Communicate findings (e.g., models or technology).
Vocabulary:	<p><u>Loop</u>: Doing something over and over again</p> <p><u>Repeat</u>: To do something again</p>
Notes:	

Week 22: Code.org, Course C, Lesson 10—Mini-Project: Sticker Art

Lesson overview:



Purpose:

This series highlights the power of loops with creative and personal designs. Offered as a project-backed sequence, this progression will allow students to build on top of their own work and create amazing artifacts.

Lesson:

- Warm Up
 - Students should have had plenty of introduction to loops at this point. Based on what you think your class could benefit from, we recommend:
 - Create a new dance with loops just like in "Getting Loopy."
 - As a class, play through a puzzle from Lesson 9, "Loops with Rey and BB-8."
 - Review how to use Artist by playing through a puzzle from "Programming in Artist."
 - Preview a puzzle from this lesson.
 - All of these options will either review loops or the artist, which will help prepare your class for fun with the online puzzles!
- Main Activity
 - Sticker Art with Loops
- Wrap Up
 - What was today's lesson about?
 - How did you feel during today's lesson?
 - What was the coolest shape or figure you programmed today? Draw it out!
 - What is another shape or figure you would like to program? Can you come up with the code to create it?

Lesson links/resources:

[Code.org, Lesson 10: Mini-Project: Sticker Art Lesson Plan](#)

CS standards addressed:

Students will be able to:

- Differentiate between commands that need to be repeated in loops and commands that should be used on their own
- Identify the benefits of using a loop structure instead of manual repetition

Standards:

- **AP.1A.2**—Model the way programs store and manipulate data by using numbers or other symbols to represent information.
- **AP.1A.3**—Develop programs with sequences and simple loops to express ideas or address a problem.
- **AP.1A.4**—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.
- **AP.1A.7**—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops

Time needed:

Total Time: 60 min

- Warm Up/Introduction **15 min**
- Main Activity **30 min**
- Wrap Up/Reflection **15 min**

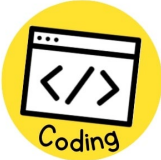
Materials needed:

Teacher:

- Computer
- Projector/smartboard with sound
- [Code.org, Lesson 10: Mini-Project: Sticker Art Lesson Plan](#)

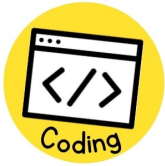
	<ul style="list-style-type: none">• Code.org account Students: <ul style="list-style-type: none">• Computer/tablet with internet access
Subject integrated:	ELA Math
Other standards addressed:	ELA <ul style="list-style-type: none">• W.2.1—Write opinion pieces in which they introduce the topic or book they are writing about, state an opinion, supply reasons that support the opinion, use linking words (e.g., because, and, also) to connect opinion and reasons, and provide a concluding statement or section. Math <ul style="list-style-type: none">• 2.G.A.1—Recognize and draw shapes having specified attributes, such as a given number of angles or a given number of equal faces.1 Identify triangles, quadrilaterals, pentagons, hexagons, and cubes.
Vocabulary:	<u>Loop</u> : The action of doing something over and over again <u>Repeat</u> : To do something again
Notes:	

Week 23: Code.org, Course C, Lesson 10—Loopy Forms and Their Functions

<p>Lesson overview:</p> 	<p>Purpose: Loopy Forms and Their Functions: Artist needs to help his friends design their house by drawing blueprints. It is important that students understand basic shapes and how each shape serves different functions in the house blueprints. They will then write opinion pieces, supply reasons, and use linking words to justify their design choices.</p> <p>Lesson:</p> <ul style="list-style-type: none"> ● Introduction <ul style="list-style-type: none"> ○ Students will work in pairs to complete Sticker Art with Loops, levels 1-8. ○ Students will draw three KWL Charts. One chart will have the topic “Triangles.” The second chart will have the topic “Quadrilaterals.” The third chart will have the topic “Pentagons.” Students will then complete the “K - Write what you know” and “W - Write what you want to know” columns of each chart. ● Main Activity <ul style="list-style-type: none"> ○ Then students will complete the Loopy Shapes & Their Functions Student Handout, using Sticker Art with Loops, levels 2, to help the Artist design house blueprints for his friends. ● Wrap Up <ul style="list-style-type: none"> ○ Have students complete the “L - Write what you learned” box on the KWL Chart.
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> ● Loopy Forms & Their Functions Lesson Plan ● Sticker Art with Loops, levels 1-8 ● Loopy Shapes & Their Functions Student Handout
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> ● Differentiate between commands that need to be repeated in loops and commands that should be used on their own ● Identify the benefits of using a loop structure instead of manual repetition <p>Standards:</p> <ul style="list-style-type: none"> ● AP.1A.2—Model the way programs store and manipulate data by using numbers or other symbols to represent information. ● AP.1A.3—Develop programs with sequences and simple loops to express ideas or address a problem. ● AP.1A.4—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions. ● AP.1A.5—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.
<p>Time needed:</p>	<p>Total Time: 60 min</p> <ul style="list-style-type: none"> ● Introduction 20 min ● Main Activity 30 min ● Wrap Up 10 min
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> ● Computer ● Projector/smartboard with sound ● Code.org, Lesson 10: Mini-Project: Sticker Art Lesson Plan ● Loopy Forms & Their Functions Lesson Plan

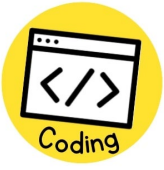
	<ul style="list-style-type: none"> • Code.org account <p>Students:</p> <ul style="list-style-type: none"> • Computer/tablet with internet access • Loopy Shapes and Their Functions Student Handout
Subject integrated:	<p>ELA Math</p>
Other standards addressed:	<p>ELA</p> <ul style="list-style-type: none"> • W.2.1—Write opinion pieces in which they introduce the topic or book they are writing about, state an opinion, supply reasons that support the opinion, use linking words (e.g., because, and, also) to connect opinion and reasons, and provide a concluding statement or section. <p>Math</p> <ul style="list-style-type: none"> • 2.G.A.1—Recognize and draw shapes having specified attributes, such as a given number of angles or a given number of equal faces. Identify triangles, quadrilaterals, pentagons, hexagons, and cubes.
Vocabulary:	<p><u>Loop</u>: The action of doing something over and over again <u>Repeat</u>: To do something again</p>
Notes:	

Week 24: Code.org, Course C, Lesson 11—The Big Event

<p>Lesson overview:</p> 	<p>Purpose: Today, students will learn to distinguish events and actions. The students will see activities interrupted by having a "button" pressed on a paper remote. When seeing this event, the class will react with a unique action. Events are widely used in programming and should be easily recognizable after this lesson.</p> <p>Lesson:</p> <ul style="list-style-type: none"> ● Warm Up <ul style="list-style-type: none"> ○ A series of events <ul style="list-style-type: none"> ■ Follow the prompts in Code.org, Lesson 11: The Big Event Lesson Plan ● Main Activity <ul style="list-style-type: none"> ○ The Big Event ● Reflection <ul style="list-style-type: none"> ○ What did we learn? ○ What are some examples of events? ○ Assessment
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> ● Code.org- Lesson 11: The Big Event Lesson Plan ● The Big Event: Assessment Answer Key ● The Big Event: Unplugged Video (Download) ● The Big Event: Assessment ● The Big Event (Course C): Event Controller
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> ● Practice differentiating pre-defined actions and event-driven ones ● Recognize movements of the teacher as signals to initiate commands ● Repeat commands given by an instructor <p>Standards:</p> <ul style="list-style-type: none"> ● AP.1A.2—Model the way programs store and manipulate data by using numbers or other symbols to represent information. ● AP.1A.4—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.
<p>Time needed:</p>	<p>Total Time: 60 min</p> <ul style="list-style-type: none"> ● Warm Up 15 min ● Main Activity 15 min ● Wrap Up 15 min ● Keyboarding Practice 15 min
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> ● Computer ● Projector/smartboard with sound ● Code.org account ● The Big Event: Assessment answer key <p>Students:</p> <ul style="list-style-type: none"> ● Computer/tablet with internet access ● The Big Event: Unplugged video (Download) ● The Big Event: Assessment ● The Big Event (Course C): Event controller
<p>Subject integrated:</p>	<p>ELA</p>
<p>Other standards</p>	<p>SL.2.1—Participate in collaborative conversations with diverse partners</p>

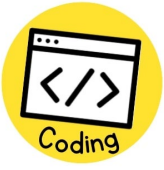
addressed:	about grade 2 topics and texts with peers and adults in small and larger groups.
Vocabulary:	<u>Event</u> : An action that causes something to happen
Notes:	

Week 25: Code.org, Course C, Lesson 12—Build a Flappy Game

<p>Lesson overview:</p> 	<p>Purpose: Events are very common in computer programs. In this lesson, students will further develop their understanding of events by making a Flappy Bird game. Students will learn to make their character move across the screen, make noises, and react to obstacles based on user-initiated events.</p> <p>Lesson:</p> <ul style="list-style-type: none"> ● Warm Up: Introduction <ul style="list-style-type: none"> ○ Review activity ● Bridging Activity <ul style="list-style-type: none"> ○ Choose 1 (paper blocks or preview online puzzle) ● Main Activity <ul style="list-style-type: none"> ○ Build a flappy game ● Wrap Up <ul style="list-style-type: none"> ○ What was today's lesson about? ○ How did you feel about today's lesson? ○ What did you do to make your game unique? ○ Draw out a game you want to make in the future?
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> ● Code.org, Lesson 12: Build A Flappy Game Lesson Plan ● Open-ended Programming Levels: Answer Key ● Unplugged Blocks (Courses C-F): Manipulatives ● Pause and Think Online: Video ● The Big Event (Course C): Event Controller
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> ● Create a game using event handlers ● Match blocks with the appropriate event handler ● Share a creative artifact with other students <p>Standards:</p> <ul style="list-style-type: none"> ● AP.1A.2—Model the way programs store and manipulate data by using numbers or other symbols to represent information. ● AP.1A.4—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.
<p>Time needed:</p>	<p>Total Time: 60 min</p> <ul style="list-style-type: none"> ● Warm Up 10 min ● Bridging Activity 10 min ● Main Activity 30 min ● Wrap Up 10 min
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> ● Computer ● Projector/smartboard with sound ● Code.org account ● Open-ended Programming Levels: Answer Key <p>Students:</p> <ul style="list-style-type: none"> ● Computer/tablet with internet access ● Unplugged Blocks (Courses C-F): Manipulatives ● Pause and Think Online: Video ● The Big Event (Course C): Event controller ● Code.org account
<p>Subject integrated:</p>	<p>ELA</p>
<p>Other standards</p>	<p>SL.2.1—Participate in collaborative conversations with diverse partners</p>

addressed:	about Grade 2 topics and texts with peers and adults in small and larger groups.
Vocabulary:	<u>Event</u> : An action that causes something to happen
Notes:	

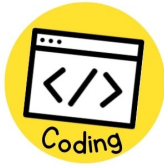
Week 26: Code.org, Course C, Lesson 12—Become a Game Designer

<p>Lesson overview:</p> 	<p>Purpose: Students will become game designers as they create and test a game. Students will also record scores, make observations, create a bar graph, analyze the results, and justify the changes they would make to their game.</p> <p>Lesson:</p> <ul style="list-style-type: none">• Vocabulary (Day 1)• Setting the Scene (Day 1)• Collecting Data (Day 1)• Switch Roles (Day 2)• Creating a Bar Graph (Day 2)• Analyzing Data (Day 3)
<p>Lesson links/resources:</p>	<ul style="list-style-type: none">• Code.org- Become a Game Designer Lesson Plan• Engineering Design Poster (display for students)• Data Collection and Analyze Sheet (print double sided)• Bar Graph Video• Create a bar graph
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none">• Create a game using event handlers• Match blocks with the appropriate event handler• Share a creative artifact with other students <p>Standards:</p> <ul style="list-style-type: none">• AP.1A.2—Model the way programs store and manipulate data by using numbers or other symbols to represent information.• AP.1A.4—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.
<p>Time needed:</p>	<p>Total Time: 60 min</p> <ul style="list-style-type: none">• Day 1 20min• Day 2 20 min• Day 3 20 min
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none">• Computer• Projector/smartboard with sound• Code.org account• Code.org, Become a Game Designer Lesson Plan• Engineering Design Poster : Display for students <p>Students:</p> <ul style="list-style-type: none">• Computer/tablet with internet access• Code.org account• Data Collection and Analyze Sheet: Print double sided• Bar Graph Video• Create a bar graph
<p>Subject integrated:</p>	<p>Math</p>
<p>Other standards addressed:</p>	<p>2.MD.10—Draw a picture graph and a bar graph (with single-unit scale) to represent a data set with up to four categories. Solve simple put-together, take-apart, and compare problems using information presented in a bar graph.</p>
<p>Vocabulary:</p>	<p><u>Game designer/programmer</u>: Someone that creates a game</p>

	<p><u>Play tester</u>: Someone that plays a game and provides feedback to the game designer</p> <p><u>Engineering design process</u>: A series of steps that guides engineering teams as they create a solution to a problem</p> <p><u>Data collection</u>: Gathering information to place in organized graphs or charts for analysis</p> <p><u>Bar graph</u>: A graph using bars to show numbers so they can be easily compared</p> <p><u>Justify</u>: To prove or show evidence supporting answers or decisions</p>
Notes:	

Week 27: Code.org, Course C, Lesson 13—Mini-Project: Chase Game

Lesson overview:



Purpose:

This lesson combines skill-building around events with a mini-project where students get to build their own animated game. Here, students will further develop their understanding of events using Play Lab. Students will use events to make characters move around the screen, make noises, and change backgrounds based on user input. At the end of the puzzle sequence, students will be presented with the opportunity to share their projects.

Lesson:

- Warm Up
 - Discuss Flappy Bird game and review events and other actions.
- Main Activity
 - Chase Game with Events
- Wrap Up
 - What was today's lesson about?
 - How did you feel about today's lesson?
 - What is an event your program used today?
 - Is there an event that you would have liked to use in your game that you did not get to use in Play Lab?

Lesson links/resources:

- [Code.org Mini-Project Lesson Plan](#)
- [Open-ended Programming Levels](#): Answer Key
- [Unplugged Blocks \(Courses C-F\)](#): Manipulatives
- [Pause and Think Online](#): Video

CS standards addressed:

Students will be able to:

- Create an animated, interactive game using sequence and event-handlers
- Identify actions that correlate to input events

Standards:

- **AP.1A.2**—Model the way programs store and manipulate data by using numbers or other symbols to represent information.
- **AP.1A.4**—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.

Time needed:

Total Time: 60 min

- Warm Up **10 min**
- Main Activity **30 min**
- Wrap Up **15 min**

Materials needed:

Teacher:

- Computer
- Projector/smartboard with sound
- [Code.org](#) account
- [Open-ended Programming Levels](#): Answer Key

Students:

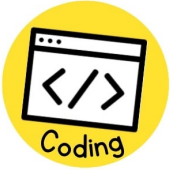
- Computer/tablet with internet access
- [Code.org](#) account
- [Unplugged Blocks \(Courses C-F\)](#): Manipulatives
- [Pause and Think Online](#): Video

Subject integrated:

ELA

Other standards addressed:	SL.2.1 —Participate in collaborative conversations with diverse partners about Grade 2 topics and texts with peers and adults in small and larger groups.
Vocabulary:	<u>Event</u> : An action that causes something to happen
Notes:	

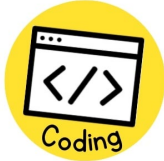
Week 28: Code.org, Course C, Lesson 14—Picturing Data

<p>Lesson overview:</p> 	<p>Purpose: Computers were created to help process data. There is an increasing amount of data in the world, so being able to read and analyze it is important. This lesson is here to make sure students have the basic experience of collecting, visualizing, and analyzing a simple set of data.</p> <p>Lesson:</p> <ul style="list-style-type: none"> ● Warm Up <ul style="list-style-type: none"> ○ The need for visualization ● Main Activity <ul style="list-style-type: none"> ○ Picturing data ● Wrap Up <ul style="list-style-type: none"> ○ Reflection
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> ● Code.org, Lesson 14: Picturing Data Lesson Plan ● Graphing Data from Play Lab : Worksheet
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> ● Collect and record data about quantities of real objects, or characters on a screen ● Create a bar graph and pie chart to represent simple data ● Make comparisons between data visualizations made by others and use them to make a prediction <p>Standards:</p> <ul style="list-style-type: none"> ● AP.1A.09—Model the way programs store and manipulate data by using numbers or other symbols to represent information. ● AP.1A.11—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.
<p>Time needed:</p>	<p>Total Time: 60 min</p> <ul style="list-style-type: none"> ● Warm Up 5 min ● Main Activity 35 min ● Wrap Up 5 min ● Keyboarding 15 min
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> ● Computer ● Projector/smartboard with sound ● Code.org account ● Code.org, Lesson 14: Picturing Data Lesson Plan <p>Students:</p> <ul style="list-style-type: none"> ● Computer/tablet with internet access ● Code.org account ● Graphing Data from Play Lab: Worksheet
<p>Subject integrated:</p>	<p>Math</p>
<p>Other standards addressed:</p>	<p>2.MD.10—Draw a picture graph and a bar graph (with single-unit scale) to represent a data set with up to four categories. Solve simple put-together, take-apart, and compare problems using information presented in a bar graph.</p>
<p>Vocabulary:</p>	<p><u>Data</u>: A collection of information</p>

Notes:

Week 29: Code.org, Course C, Lesson 15—Binary Bracelets

Lesson overview:



Purpose:

In this lesson students will learn how information is represented in such a way that a computer can interpret and store it. When learning binary, students will have the opportunity to write codes and share them with peers as secret messages. This can then be related back to how computers read a program, translate it to binary, use the information in some way, then reply back in a way humans can understand. For example, when we type a sentence into a document then press save, a computer translates the sentence into binary, stores the information, then posts a message indicating the document has been saved.

Lesson:

- Warm Up
 - Introduce vocabulary
 - [Binary Bracelets](#): Lesson in action video
 - Off and on
 - Remarks
- Main Activity
 - [Binary Bracelets](#): Worksheet
 - [Bits Versus Bytes](#): Student video
- Wrap Up
 - Flash chat:
 - What else do you think is represented as binary inside of a computer?
 - How else might you represent binary instead of boxes that are filled or not filled?
 - What was your favorite part about that activity?
 - Prompts:
 - What was today's lesson about?
 - How else might you represent binary instead of boxes that are filled or not filled?
 - What was your favorite part about that activity?
- Assessment
 - [Binary Bracelets](#): Assessment

Lesson links/resources:

- [Code.org: Binary Bracelets Lesson Plan](#)
- [Binary Bracelets](#): Lesson in Action Video
- [Binary Bracelets](#): Assessment Answer Key
- [Binary Bracelets](#): Unplugged Video ([Download](#))
- [Binary Bracelets](#): Worksheet
- [Binary Bracelets](#): Assessment
- [Bits Versus Bytes](#): Student Video

CS standards addressed:

Students will be able to:

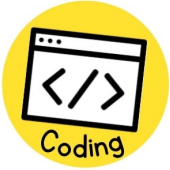
- Decode binary back to letters
- Encode letters into binary
- Relate the idea of storing letters on paper to the idea of storing information in a computer

Standards:

- **AP.1A.09**—Model the way programs store and manipulate data by using numbers or other symbols to represent information.
- **AP.1A.11**—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.

Time needed:	<p>Total Time: 65 min</p> <ul style="list-style-type: none"> ● Warm Up 15 min ● Main Activity 20 min ● Wrap Up 5 min ● Assessment 15 min
Materials needed:	<p>Teacher:</p> <ul style="list-style-type: none"> ● Computer ● Projector/smartboard with sound ● Code.org account ● Code.org: Binary Bracelets Lesson Plan ● Binary Bracelets: Lesson in Action Video ● Binary Bracelets: Assessment answer key <p>Students:</p> <ul style="list-style-type: none"> ● Computer/tablet with internet access ● Code.org account ● Binary Bracelets: Unplugged video (Download) ● Binary Bracelets: Worksheet ● Binary Bracelets: Assessment ● Bits Versus Bytes: Student video
Subject integrated:	ELA
Other standards addressed:	SL.2.6 —Produce complete sentences when appropriate to task and situation in order to provide requested detail or clarification.
Vocabulary:	<u>Binary</u> : A way of representing information using only two options
Notes:	

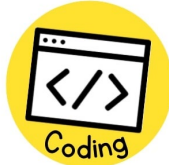
Week 30: Code.org, Course C, Lesson 16—End of Course Project

<p>Lesson overview:</p> 	<p>Purpose: This lesson provides students with space to create a project of their own design, using a step-by-step process that requires planning but also allows for broad creativity.</p> <p>Lesson:</p> <ul style="list-style-type: none">• Warm Up<ul style="list-style-type: none">◦ Play Lab Project Planning Guide: Worksheet• Main Activity<ul style="list-style-type: none">◦ Coding project• Wrap Up<ul style="list-style-type: none">◦ Showcase
<p>Lesson links/resources:</p>	<ul style="list-style-type: none">• Code.org, Lesson 16: End of Course Project• Play Lab Project Planning Guide (Example): Lesson Resource• Play Lab Project Planning Guide: Worksheet
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none">• Overcome obstacles such as time constraints or bugs• Use a planned design as a blueprint for creation <p>Standards:</p> <ul style="list-style-type: none">• AP.1A.10—Develop programs with sequences and simple loops, to express ideas or address a problem.• AP.1A.12—Develop plans that describe a program's sequence of events, goals, and expected outcomes.• AP.1A.13—Give attribution when using the ideas and creations of others while developing programs.• AP.1A.15—Using correct terminology, describe steps taken and choices made during the iterative process of program development.
<p>Time needed:</p>	<p>Total Time: 60 min</p> <ul style="list-style-type: none">• Warm Up 10 min• Main Activity 25 min• Wrap Up 10 min• Keyboarding 15 min
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none">• Computer• Projector/smartboard with sound• Code.org account• Code.org, Lesson 16: End of Course Project• Play Lab Project Planning Guide (Exemplar): Lesson Resource <p>Students:</p> <ul style="list-style-type: none">• Computer/tablet with internet access• Code.org account• Play Lab Project Planning Guide: Worksheet
<p>Subject integrated:</p>	<p>ELA</p>
<p>Other standards addressed:</p>	<p>SL.2.3—Ask and answer questions about what a speaker says in order to clarify comprehension, gather additional information, or deepen understanding of a topic or issue.</p>

Vocabulary:	<u>Actor</u> : A participant in an action or process
Notes:	

Week 31: Gingerbread Coding

Lesson overview:



Purpose:

[Gingerbread Coding](#) (Brooke Brown, *Teach Outside the Box*) is a lesson where students will use block coding to create a path for the gingerbread man to escape.

Lesson:

- Partner students. Each pair of students will need one Gingerbread Coding Mat (page 3), one set of Map Pieces (Colored on pages 4-5 or black and white on pages 6-7), and 2-4 copies of “Crack the Code!” (Page 8). You may also choose to put copies of page 8 inside clear page protectors so that students can write and wipe codes with dry erase markers multiple times. Page 9 is optional and is provided for you to project or display coding symbols.
- Have pairs of students cut out all the map pieces and color if desired.
- Student 1 arranges the map pieces on the Gingerbread Coding Map, starting with the gingerbread man or woman and ending with the Gingerbread House, with path pieces (colored squares) in between to connect them. Then he or she places two Treats and one Enemy along the path.
- Student 2 then “codes” the path of the gingerbread man or woman on page 8, using the provided symbols to draw the directions that he or she must travel. When the gingerbread man or woman comes to a treat, they will draw the symbol to “eat the treat,” and when they reach an enemy, they draw the symbol to “jump over the enemy.” (Encourage students to use cardinal directions when referring to directions.)
- Student 1 checks the code and coaches Student 2 as needed.
- Map pieces are cleared and students trade places, with Student 2 creating the map and Student 1 coding the action.

Lesson links/resources:

[Gingerbread Coding](#)

CS standards addressed:

Students will be able to:

- Use basic coding to accomplish tasks
- Debug errors in their codes

Standards:

- **AP.1A.1**—Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.
- **AP.1A.3**—Develop programs with sequences and simple loops to express ideas or address a problem.

Time needed:

70 min

- Introduction to the activity **10 min**
- Gingerbread Coding activity **50 min**
- Closing **10 min**

Materials needed:

Teacher:

- Computer
- Projector/smartboard with sound
- [Teachers Pay Teachers](#) account

Students:

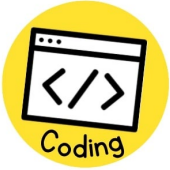

- [Gingerbread Coding](#)

Subject integrated:

Social Studies

Other standards addressed:	G.2.3.2 —Identify cardinal and intermediate directions (e.g., north, northeast, northwest, southwest, southwest, east, and west.)
Vocabulary:	
Notes:	→Teachers will need to create FREE teacher account for Teachers Pay Teachers

Week 32: Kodable—Pizza Party

<p>Lesson overview:</p>  <p style="text-align: center;">Coding</p>  <p style="text-align: center;">Unplugged</p>	<p>Purpose: Students will practice creating a sequence. Students will be able to write simple numerical expressions and evaluate them in proper sequence.</p> <p>Lesson:</p> <ul style="list-style-type: none"> • The students will build a sequence in code to create their slices of pizza. Each day consists of a 20-25 min session. <ul style="list-style-type: none"> ○ Day 1: <ul style="list-style-type: none"> ■ Vocabulary, ■ I follow a sequence when I..., ■ Exit Ticket ○ Day 2: <ul style="list-style-type: none"> ■ Brush Teeth Algorithm, ■ Make a Pizza ■ Exit Ticket ○ Day 3: <ul style="list-style-type: none"> ■ Inquiry Sheet ■ Exit Ticket
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> • Kodable Lesson Frame • Pizza Party Worksheets • Dominic's Pizza Party • Sequence Sector lesson 1: "1, 2, 3 Roll" 1.1-1.5 Optional On-Screen Practice
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> • Create and follow algorithms • Break down the steps needed to solve a problem • Develop a plan to illustrate what a program will do <p>Standards:</p> <ul style="list-style-type: none"> • AP.1A.1—Model daily processes by creating and following algorithms. • AP.1A.4—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions. • AP.1A.5—Develop plans that describe a program's sequence of events, goals, and expected outcomes.
<p>Time needed:</p>	<p>Total Time: 55-65 min This lesson can be broken down into 3 days as indicated in the Kodable lesson framework:</p> <p>Day 1 (20-25 min)</p> <ul style="list-style-type: none"> • Vocabulary cards with visuals and definitions 5-10 min • Sequence sentence frame graphic organizer 10 min • Exit ticket 5 min <p>Day 2 (20 min)</p> <ul style="list-style-type: none"> • Dominic's Pizza video 2 min • "Brush teeth" algorithm pseudocode visual 3 min • Pizza ingredients (build your own!) 10 min • Exit ticket: School algorithm 5 min <p>Day 3 (15-20 min)</p> <ul style="list-style-type: none"> • Inquiry sheet 5 min • Exit ticket: K-W-L 5 min • Kodable on-screen practice 5-10 min
<p>Materials needed:</p>	<p>For the students:</p> <ul style="list-style-type: none"> • Pizza Party Worksheets

	<ul style="list-style-type: none"> ● Vocabulary cards with visuals and definitions ● Sequence sentence frame graphic organizer ● Exit ticket: Emoji ● Dominic's Pizza video ● "Brush teeth" algorithm pseudocode visual ● Pizza algorithm pseudocode graphic organizer ● Pizza ingredients (build your own!) ● Exit ticket: School algorithm ● Inquiry sheet ● Exit ticket: K-W-L ● Optional: Kodable on-screen practice
Subject integrated:	ELA
Other standards addressed:	<p>ELA</p> <ul style="list-style-type: none"> ● SL.2.1—Participate in collaborative conversations with diverse partners about Grade 2 topics and texts with peers and adults in small and larger groups. ● SL.2.2—Recount or describe key ideas or details from a text read aloud or information presented orally or through other media. ● SL.2.3—Ask and answer questions about what a speaker says in order to clarify comprehension, gather additional information, or deepen understanding of a topic or issue.
Vocabulary:	<p><u>Programmer</u>: A person who writes code and communicates instructions to a computer</p> <p><u>Program</u>: A sequence of instructions given to a computer in code that a computer can understand. The computer follows the instructions and carries out the task.</p> <p><u>Language</u>: A way of communicating ideas or feelings through sounds, symbols, signs, or words. There are thousands of languages in the world.</p> <p><u>Communication</u>: The act of using words, signs, sounds, or symbols to exchange information to someone else</p> <p><u>Code</u>: The language written by humans that gives instructions to a computer</p> <p><u>Command</u>: A specific instruction given to a computer in written code from a programmer</p> <p><u>Sequence</u>: Instructions given to the computer to be followed in the exact order they are written in code using commands from programmers</p> <p><u>Algorithm</u>: A sequence of steps followed in order to finish a task (can be performed with or without a computer)</p> <p><u>Bugs</u>: Mistakes in the program's code that cause the program to function in a way that was not intended</p> <p><u>Debugging</u>: The process of finding and fixing mistakes in the program so it will run properly</p>
Notes:	<p>This lesson is developed to take place over the course of 3 days (20-25 min lesson each.) You can modify it to take place on one day if needed.</p> <p>→Teachers will need to create a FREE teacher account for Kodable.</p>

Week 33: Kodable—Magnificent Maze Maker

Lesson overview:	<p>Purpose: Can you build a symmetrical maze? A maze with right angles? Get creative and complete maze-building challenges with basic coding concepts. Students will be able to create solvable mazes while applying grade-level geometry concepts. Students can create a maze from one place to another, such as mapping from the classroom to the library.</p> <p>Lesson:</p> <ul style="list-style-type: none"> ● Introduction <ul style="list-style-type: none"> ○ Follow prompts ○ Model how to create maze by showing Make Your Own Kodable Maze ○ Review maze maker tips ● Maze Maker Challenge ● Creative Challenge (Level 6) <ul style="list-style-type: none"> ○ Let students spend time on this level being creative! They must create and solve their puzzle before they can swap with another student. ● Share <ul style="list-style-type: none"> ○ Let students swap seats (or devices) with other students to attempt each other's puzzles.
Lesson links/resources:	<ul style="list-style-type: none"> ● Kodable Magnificent Maze Maker Lesson Plan ● Reproducible ● Make Your Own Kodable Maze ● Kodable.com Magnificent Maze maker
CS standards addressed:	<p>Students will be able to:</p> <ul style="list-style-type: none"> ● Students will be able to create solvable mazes while applying grade-level geometry concepts <p>Standards:</p> <ul style="list-style-type: none"> ● AP.1A.1—Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.
Time needed:	<p>Total Time: 50-60 min</p> <ul style="list-style-type: none"> ● Introduction 10 min ● Maze Maker Challenge 10-20 min ● Creative Challenge 15 min ● Share 15 min
Materials needed:	<p>Teacher:</p> <ul style="list-style-type: none"> ● Computer ● Projector/smartboard with sound ● Kodable Magnificent Maze Maker Lesson Plan ● Make Your Own Kodable Maze <p>Students:</p> <ul style="list-style-type: none"> ● Computer/tablet with internet access ● Kodable.com Magnificent Maze maker ● Reproducible
Subject integrated:	Math
Other standards addressed:	2.G.2 —Partition a rectangle into rows and columns of same-size squares and count to find the total number of them.

Vocabulary:	
Notes:	

Week 34: Kodable—Magnificent Maze Maker Extended and Unplugged

Lesson overview:



Purpose:

Students will use their knowledge of coding to create a symmetrical marble maze using Legos and Lego base plates

Students can create a maze from one place to another such as mapping from the classroom to the library.

Lesson:

- Introduction
 - Review the Magnificent Maze Maker lesson from the previous week.
 - Introduce this week's lesson as the unplugged option.
 - Show the [How to make a Lego maze video](#)
- Magnificent Maze Maker Extended (Unplugged)
 - Students will each need a Lego plate, Legos, and marbles for this activity.
 - Students can work alone, or they can work in pairs.
 - Students should sketch a map of the path they would take from the classroom to a nearby location (ex: restroom, office, gym, lunchroom, library, etc.)
 - Once they sketch the map, students should create that map, using Legos, on the Lego plate.
 - Then, students should write the code a marble would need to take to reach the destination (just like in the Kodable lesson from last week.)
- Wrap Up
 - What did you like most about today's lesson?
 - What did you like least about today's lesson?
 - What did you learn today during this lesson?

Lesson links/resources:

[How to make a Lego maze video](#)

CS standards addressed:

Students will be able to:

- Students will be able to create solvable mazes while applying grade-level social studies concepts

Standards:

- **AP.1A.1**—Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.

Time needed:

Total Time: 60 min

- Introduction **5 min**
- Magnificent Maze Maker Extended **40 min**
- Wrap Up **5 min**

Materials needed:

Teachers:

- [How to make a Lego maze video](#)

Students:


- Lego plates
- Legos
- Marbles
- Paper
- Pencil/pen

Subject integrated:

Math
Social Studies

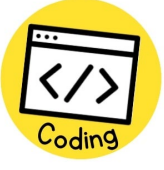
Other standards addressed:	Math <ul style="list-style-type: none">• 2.G.2—Partition a rectangle into rows and columns of same-size squares and count to find the total number of them. Social Studies <ul style="list-style-type: none">• G.2.3.2—Identify cardinal and intermediate directions (e.g., north, northeast, northwest, southwest, southwest, east, and west.)
Vocabulary:	
Notes:	

Week 35: Kodable—Introduction to Coding and Coding Basics Unplugged

<p>Lesson overview:</p> 	<p>Purpose: Students will review basic programming skills and practice using core coding commands.</p> <p>Lesson:</p> <ul style="list-style-type: none"> • Unplugged Activity <ul style="list-style-type: none"> ◦ Write algorithm for brushing teeth one day, practice the sequence to look for bugs and debug the next day, then compare the strategies and talk about successes. • Keyboarding <ul style="list-style-type: none"> ◦ Choose an online or unplugged keyboard practice game. • Introduce the definition of code (#4 under “Introduction to Hour of Code” section of lesson). <ul style="list-style-type: none"> ◦ What Are Computer Programs? video • Unplugged Activity <ul style="list-style-type: none"> ◦ My First Code Worksheet Packet • Wrap Up <ul style="list-style-type: none"> ◦ Debrief and reflect ◦ (DEBRIEF) “Earlier I asked, ‘What does a programmer or a computer scientist do?’. Now, after watching the video and working on our packets, what do you think? What does a programmer do?” ◦ (REFLECT) “What was difficult about this packet? What was your favorite worksheet in the packet? What is something you did, or learned, from today’s Hour of Code that you want to share with your family?” (Move to overview or time.)
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> • Kodable: Coding Basics: Unplugged • My First Code Worksheet Packet
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> • Create and model algorithms • Express ideas or address problems by developing programs with a sequence • Break down the steps needed to solve a problem into a precise sequence of instructions • Use various strategies to fix problems in algorithms <p>Student Objectives:</p> <ul style="list-style-type: none"> • AP.1A.1—Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks. • AP.1A.3—Develop programs with sequences and simple loops to express ideas or address a problem. • AP.1A.4—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions. • AP.1A.7—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.
<p>Time needed:</p>	<p>Total Time: 50 min</p> <ul style="list-style-type: none"> • Keyboarding 15 min • Kodable Lesson (35 min) <ul style="list-style-type: none"> ◦ Introduction to Coding 5 min ◦ Unplugged Activity: 20 min ◦ Wrap Up 10 min

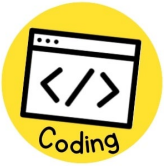
Materials needed:	<p>Teacher:</p> <ul style="list-style-type: none"> • Computer • Projector/smartboard with sound • Kodable account • Kodable, Coding Basics: Unplugged <p>Students:</p> <ul style="list-style-type: none"> • Computer/Tablet with internet access • Kodable account • Kodable Unplugged Worksheet Packet • Intro to Coding for Kids 4 -6 Video • Pen/pencil
Subject integrated:	ELA
Other standards addressed:	<ul style="list-style-type: none"> • RL.2.1—Ask and answer such questions as who, what, where, when, why, and how to demonstrate understanding of key details in a text. • SL.2.1—Participate in collaborative conversations with diverse partners about Grade 2 topics and texts with peers and adults in small and larger groups.
Vocabulary:	<p><u>Code</u>: The programming language that humans create and use to tell computers what to do</p> <p><u>Programmer</u>: A person who writes the code (language) that tells the computer what to do</p> <p><u>Sequence</u>: An order of events</p> <p><u>Condition</u>: An exception to a rule. Conditional statements are “If, then” statements: If a condition is true, then something happens.</p> <p><u>Loop</u>: A command used to repeat a portion of code</p> <p><u>Function</u>: A set of steps given a simple name that a programmer can easily call on and reuse again in a program</p> <p><u>Variable</u>: A container that stores a value that can change</p>
Notes:	<p>→Teachers will need to create FREE teacher and/or student accounts (when applicable) at https://www.kodable.com/</p>

Week 36: Kodable—Build Your Own Fuzz!

<p>Lesson overview:</p> 	<p>Purpose: What are some character traits you can identify in books or people you know? Design a fuzz and then modify properties to build it.</p> <p>Lesson:</p> <ul style="list-style-type: none"> • Introduction <ul style="list-style-type: none"> ◦ Review characters and character traits. (Give examples from books, movies, games.) ◦ Explain that today we will focus on game characters. ◦ Show Meet blueFuzz! Video • Main Activity <ul style="list-style-type: none"> ◦ Build Your Own Fuzz <ul style="list-style-type: none"> ■ Use a graphic organizer to design fuzz, then complete Build Your Own Fuzz! On Kodable. • Wrap Up <ul style="list-style-type: none"> ◦ Let students share their fuzz with classmates, allowing students to describe character traits.
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> • Kodable: Build Your Own Fuzz Lesson Plan • Meet blueFuzz! Video • Build Your Own Fuzz! Student Worksheet
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> • Modify JavaScript properties to customize a fuzz character <p>Standards:</p> <ul style="list-style-type: none"> • CS.1A.1—Select and operate appropriate software to perform a variety of tasks and recognize that users have different needs and preferences for technology they use.
<p>Time needed:</p>	<p>Total Time: 50 min</p> <ul style="list-style-type: none"> • Introduction 10 min • Main Activity 30 min • Wrap Up 10 min
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> • Computer • Projector/smartboard with sound • Kodable account • Kodable: Build Your Own Fuzz Lesson Plan • Meet blueFuzz! Video <p>Students:</p> <ul style="list-style-type: none"> • Computer/tablet with internet access • Kodable account • Build Your Own Fuzz! Student Worksheet
<p>Subject integrated:</p>	<p>ELA</p>
<p>Other standards addressed:</p>	<p>RL.2.7—Use information gained from the illustrations and words in a print or digital text to demonstrate understanding of its characters, setting, or plot.</p>
<p>Vocabulary:</p>	<p><u>Variable</u>: A container that stores a value that can change <u>Value</u>: Information stored in a variable. Values can be written as text (strings) or numbers (integers). <u>Properties</u>: Features that describe an object</p>

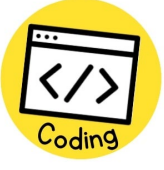
Notes:

Week 37: Kodable—If Flash, Then Clap!

<p>Lesson overview:</p> 	<p>Purpose: Students will explore conditional statements using critical thinking skills and incorporating science.</p> <p>Lesson:</p> <ul style="list-style-type: none">• Direct Instruction<ul style="list-style-type: none">◦ Guidance provided on lesson plan• If Lightning, then Thunder!<ul style="list-style-type: none">◦ A storm is coming! Students examine the logic behind thunder and lightning and relate this condition to programming.
<p>Lesson links/resources:</p>	<ul style="list-style-type: none">• If Flash, then Clap! Lesson Plan• Lesson Resources• If Statements Video• What Causes Thunder and Lightning Video
<p>CS standards addressed:</p>	<p>Student will be able to:</p> <ul style="list-style-type: none">• Students will be able to determine the effect of a condition being true• Students will be able to connect real-world conditions with “if statements” in programming• Students will be able to create “if statements” to describe real world cause and effects <p>Standards:</p> <ul style="list-style-type: none">• AP.1A.4—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.
<p>Time needed:</p>	<p>Total Time: 50 min</p> <ul style="list-style-type: none">• Direct Instruction 15 min• If Lightning, then Thunder! 35 min• Optional On-screen Practice 10 min
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none">• Computer• Projector/smartboard with sound• Kodable account• If Flash, then Clap! Lesson Plan• Lesson Resources• If Statements Video• What Causes Thunder and Lightning Video <p>Students:</p> <ul style="list-style-type: none">• Computer/tablet with internet access• Kodable account• Code.org Video: Conditional Statements• Video: What is Thunder and Lightning?• Decision tree graphic organizer• Vocabulary cards• 1 Fluorescent light bulb• 1 Rubber balloon• Brown paper bags for each student• Exit ticket: Weather If statements
<p>Subject integrated:</p>	<p>Math</p>

Other standards addressed:	2.OA.2 —Fluently add and subtract within 20 using mental strategies. By the end of Grade 2, know from memory all sums of two one-digit numbers.
Vocabulary:	<p><u>Sequence</u>: Instructions given to the computer to be followed in the exact order they are written. Written in code using commands from programmers.</p> <p><u>Condition</u>: Allows the program to perform different actions, depending on the condition being true or false</p> <p><u>Program</u>: A sequence of instructions given to a computer in code that a computer can understand. The computer follows the instructions and carries out the task.</p> <p><u>Programmer</u>: A person who writes code and communicates instructions to a computer</p> <p><u>Code</u>: The language written by humans that gives instructions to a computer</p> <p><u>Command</u>: A specific instruction given to a computer in written code from a programmer</p> <p><u>If statement</u>: A logic statement used in programming. Allows a computer program to act differently each time it is executed, depending on if an input is evaluated to be either true or false.</p>
Notes:	

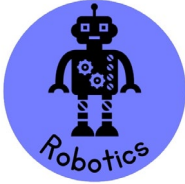
Week 38: Kodable—Beach Clean Up

<p>Lesson overview:</p>  <p>Coding</p>	<p>Purpose: Students will be able to design and create mazes based on preexisting obstacles. Students will then write simple programs to solve mazes using basic coding concepts. Next the students will examine ways technology can be used to solve real-world problems.</p> <p>Lesson:</p> <ul style="list-style-type: none">• Introduction<ul style="list-style-type: none">◦ Students will watch the Tommy the SudBudz Turtle - Listen Up Yo! Keep the Oceans Clean ;) video.• Brainstorm<ul style="list-style-type: none">◦ The teacher will talk about types of pollution, and students will fill out the “Brainstorm” section of the Lesson Worksheet.• Beach Cleanup<ul style="list-style-type: none">◦ The students will complete the Beach Cleanup coding activity.◦ Note: Students will need to create the path all the way across the screen in order to be able to play their game.• Wrap Up<ul style="list-style-type: none">◦ Students will complete the wrap up portion of the Lesson Worksheet.
<p>Lesson links/resources:</p>	<ul style="list-style-type: none">• Kodable Lesson Frame• Beach Cleanup• Lesson Worksheet• Tommy the SudBudz Turtle - Listen Up Yo! Keep the Oceans Clean ;)
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none">• Students will be able to design and create mazes based on preexisting obstacles• Students will be able to write simple programs to solve mazes using basic coding concepts• Students will be able to examine ways technology can be used to solve real-world problems• Students will be able to collaborate and communicate effectively with peers <p>Standards:</p> <ul style="list-style-type: none">• AP.1A.3—Develop programs with sequences and simple loops to express ideas or address a problem.• AP.1A.4—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.• AP.1A.7—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.
<p>Time needed:</p>	<p>Total Time: 46-60 min</p> <ul style="list-style-type: none">• Intro Video 6 min• Group Brainstorm 15 min• Coding Activity 15 min• Wrap Up 10 min
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none">• Computer• Projector/smartboard with sound• Kodable account• Kodable Lesson Frame• Tommy the SudBudz Turtle - Listen Up Yo! Keep the Oceans Clean ;)

	<p>Students:</p> <ul style="list-style-type: none"> • Computer/tablet with internet access • Kodable account • Beach Cleanup • Lesson Worksheet • Pens/pencils
<p>Subject integrated:</p>	<p>ELA Science</p>
<p>Other standards addressed:</p>	<p>ELA</p> <ul style="list-style-type: none"> • SL.2.1—Participate in collaborative conversations with diverse partners about Grade 2 topics and texts with peers and adults in small and larger groups. • SL.2.2—Recount or describe key ideas or details from a text read aloud or information presented orally or through other media. • SL.2.3—Ask and answer questions about what a speaker says in order to clarify comprehension, gather additional information, or deepen understanding of a topic or issue. <p>Science</p> <ul style="list-style-type: none"> • E.2.10.1—Use informational text, other media, and first-hand observations to investigate, analyze and compare the properties of Earth materials (including rocks, soils, sand, and water.)
<p>Vocabulary:</p>	<p><u>Program</u>: Step-by-step instructions written in programming language (code) that a computer can read and follow <u>Sequence</u>: An order of events executed by a computer exactly as it is written <u>Condition</u>: A statement that tells a program to run in a certain way, only if certain conditions are met. Conditional statements are “If, then” statements: If a condition is true, then something happens. <u>Loop</u>: A command used to repeat a portion of code until a desired process is complete <u>Function</u>: A set of steps given a simple name that a programmer can easily call on and re-use again in a program <u>Debug</u>: The process of finding and fixing bugs (mistakes) in code so that the computer program will run as expected</p>
<p>Notes:</p>	

Week 39: Spelling Robotics

Lesson overview:



Purpose:

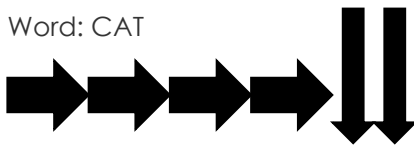
In this lesson, students will spell a word correctly in order to gather cheese. After a student spells the given word correctly, he/she will write a code to gather cheese. The student that gathers three pieces of cheese is the winner.

Lesson:

- Introduction
 - Explain to students the function of the codable robot.
 - Discuss the purpose of the lesson and the objective.
- Setup
 - Students can participate individually or in groups.
 - Teachers will write individual letters of the alphabet on index cards.
 - The teacher will create a grid for the robot to travel.
 - The teacher can use floor tiles as the grid or tape off a particular area.
 - Teacher will designate the top right square of the grid as the start block.
 - Teachers will place the letters of spelling words in the tiles.
- Main Activity
 - Teachers will call out the word and have students start by spelling the word correctly.
 - Students can use a white board or pencil/paper to spell the word correctly.
 - Next, students will use [coding cards](#) to map out the code they will use to program the robot to travel through the blocks and collect the correct letters (in order).
 - Students will need to code the robot (Some examples include: [Code and Go Mouse](#), [Botley](#), [Dash](#)) to travel through the letters in order to spell the high frequency word.
 - If you do not have a coding robot in your class, the students can act as the robot.

Example:

Word: CAT



START		C		A
	K		O	
I				T

- Exit Ticket
 - What was your favorite part of today's activity?
 - What words did you spell correctly?
 - What are some words you spelled incorrectly?

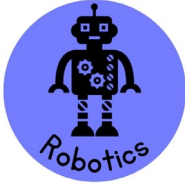
Lesson links/resources:

- Codable Robot (Some examples include: [Code and Go Mouse](#), [Botley](#), [Dash](#))
- [coding cards](#)

CS standards addressed:	<p>The students will be able to:</p> <ul style="list-style-type: none"> • Code a robot correctly to reach a destination <p>Standards:</p> <ul style="list-style-type: none"> • AP.1A.4—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions. • AP.1A.5—Develop plans that describe a program's sequence of events, goals, and expected outcomes.
Time needed:	<p>Total Time: 60 min</p> <ul style="list-style-type: none"> • Review 10 min • Main Activity 40 min • Exit Ticket 10 min
Materials needed:	<p>Teachers:</p> <ul style="list-style-type: none"> • Alphabet cards (each card should have individual letters-teacher will create) • 175 Second Grade Spelling Words <p>For the students:</p> <ul style="list-style-type: none"> • Robot (Some examples include: Code and Go Mouse, Botley, Dash) • Coding cards
Subject integrated:	ELA
Other standards addressed:	RF.2.3 —Know and apply grade-level phonics and word analysis skills in decoding words.
Vocabulary:	<p><u>Algorithm</u>: A list of steps to finish a task <u>Bug</u>: Part of a program that does not work correctly <u>Debugging</u>: Finding and fixing problems in an algorithm or program <u>Sequencing</u>: Putting commands in correct order to understand commands</p>
Notes:	

Week 40: Marching Orders

Lesson overview:

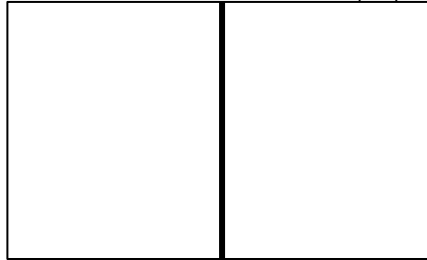


Purpose:

Students sit back-to-back. One student gives verbal directions while the other student follows the given directions.

Lesson:

- Teacher Preparation
 - Watch [Unplugged Coding Activity Video](#) to better understand how the activity will take place.
- Introduction
 - Provide students with a copy of shapes that your students should know.
 - Ex: [Quadrilateral/Non-Quadrilateral Shapes](#)
 - Show the [Learning Shapes 2nd Grade Video](#)
- Main Activity
 - Students will be paired up, and they will sit back-to-back.
 - One student will have a copy of shapes, while the other has a pen and pencil and a piece of copy paper that has a line drawn in the middle of the paper as pictured below.



- The student with the shapes should give directions to the student with pencil and paper on how to draw the shape step by step.
 - The student drawing should start on the left side of the paper for the first attempt.
 - The students should not identify the shape, tell how many sides it has, give any hints to what the shape is, etc.
 - Watch the video under teacher preparation to better understand what is expected.
 - Ex: Triangle
 - Students giving directions may say, "Start in the middle and draw a line across, then draw a line up to the left, then draw a line down."
 - The first drawing may look nothing like a triangle, but that is okay!
 - Have the student giving directions look at the shape that was drawn to determine if it is correct. (More than likely, it will not be correct.)
 - The student giving directions should think of more specific instructions to provide the artist with more detail and try the activity again on the right side of the paper.
- The goal of this activity is for students to learn how to give step by step directions (algorithm), find bugs, and debug a program.
- Have students swap roles, and the new artist will attempt to follow the directions given by the other student.
- Repeat several times.

	<ul style="list-style-type: none"> ● Wrap Up: <ul style="list-style-type: none"> ○ What was the most difficult part of this activity? ○ What was your favorite part of this activity? ○ What did you learn from this activity?
Lesson links/resources:	<ul style="list-style-type: none"> ● Unplugged Coding Activity Video ● Learning Shapes 2nd Grade Video ● Quadrilateral/Non-Quadrilateral Shapes
CS standards addressed:	<p>Students will be able to:</p> <ul style="list-style-type: none"> ● Give step by step directions (algorithm), find bugs, and debug a program <p>Standards:</p> <ul style="list-style-type: none"> ● AP.1A.3—Develop programs with sequences and simple loops to express ideas or address a problem.
Time needed:	<p><u>Total Time: 55 min</u></p> <ul style="list-style-type: none"> ● Introduction 5 min ● Main Activity 40 min ● Wrap Up 10 min
Materials needed:	<p>Teachers:</p> <ul style="list-style-type: none"> ● Computer ● Projector/smartboard with sound ● Unplugged Coding Activity Video ● Learning Shapes 2nd Grade Video <p>Students:</p> <ul style="list-style-type: none"> ● Pen/pencil ● Copy paper (with line drawn in middle) ● Quadrilateral/Non-Quadrilateral Shapes (or other resource with shapes)
Subject integrated:	Math
Other standards addressed:	2.G.1 —Recognize and draw shapes having specified attributes, such as a given number of angles or a given number of equal faces. Identify triangles, quadrilaterals, pentagons, hexagons, and cubes.
Vocabulary:	<p><u>Algorithm</u>: A list of steps to finish a task</p> <p><u>Sequencing</u>: Putting commands in correct order to understand commands</p>
Notes:	

Appendix A: Code.org

I'd like to start using Code.org in my classroom. How should I start?

<https://support.code.org/hc/en-us/articles/228116468-I-d-like-to-start-using-Code-org-in-my-classroom-How-should-I-start->

How to create a teacher account:

<https://support.code.org/hc/en-us/articles/228116468-I-d-like-to-start-using-Code-org-in-my-classroom-How-should-I-start->

How to create a classroom section:

<https://support.code.org/hc/en-us/articles/115000488132-Creating-a-classroom-section>

Finding curriculum and lesson plans:

<https://support.code.org/hc/en-us/articles/115001595051-Finding-curriculum-and-lesson-plans>

Code.org Support

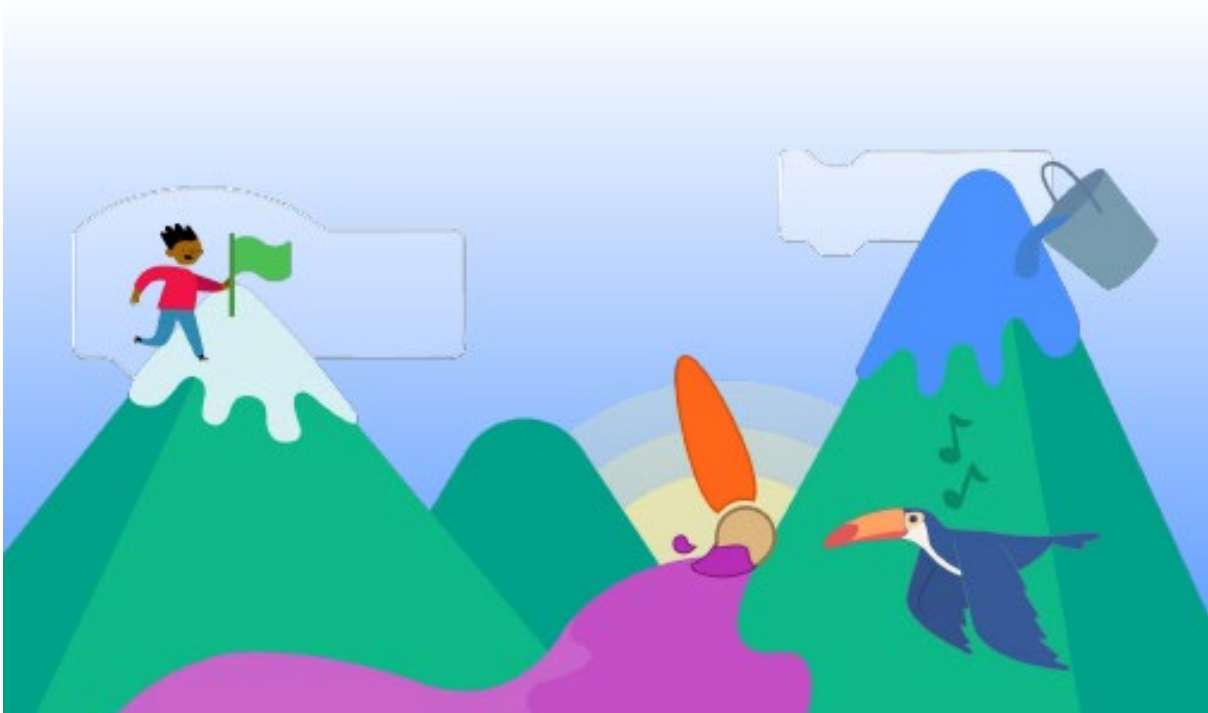
<https://support.code.org/hc/en-us>

Appendix B: Scratch

SCRATCH

Educator's Guide

- Teacher Accounts
- Beginner's Guide
- Lesson Guides





Teacher Accounts

As an educator, you can request a Scratch Teacher Account. A Scratch Teacher Account provides educators with additional features to manage student participation on Scratch, including the ability to create student accounts, organize student projects into studios, and monitor student comments. This guide will walk you through creating an account, creating a class, adding and managing your students, and creating class studios. You can also see our [Scratch for Educators](#) page and our [Teacher Account FAQ](#) page for additional information.

Create Your Teacher Account

Visit this link to get started: <https://scratch.mit.edu/educators/register>

You'll be prompted to create a username and password. **Make sure that your username does not contain your name or personal information**, like your school, location, or email address.

Within the Scratch community, all users are asked to refrain from sharing personal information through their usernames. **It's important that both you and your students follow these guidelines. Accounts that do not adhere to these guidelines will be deleted.**

Creating your teacher account

Create a username

QuirkyArtTeacher

Password

.....

Show password

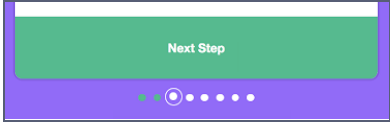


Next Step

Tips for making your username

- **Incorporate the name of the subject you teach**
 - ex: QuirkyArtTeacher
- **Use a tool or term from the subject you teach**
 - ex: MetamorphicRocks
- **Add an important date, be unique**
 - ex: Bibliophile1440
- **Make it memorable with a pun or an alliteration!**
 - ex: TyranoTeacher

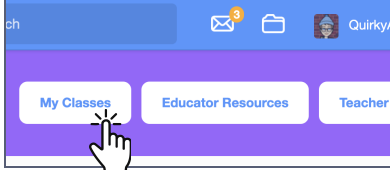

Be sure to make a note of your username and password.



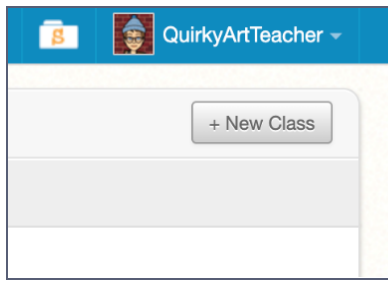
	<p>Click through each step to complete registration.</p>
	<p>Log into your email and confirm your email address.</p> <p>Check your spam folder if you do not see the email.</p> <p>Once you have confirmed your email address, we'll review your account.</p>
	<p>Once your account has been reviewed and approved, you will receive a welcome email. Then, you can log into your teacher account at scratch.mit.edu!</p>

Create a Class

Creating classes allows you to manage groups of students, and create studios where your students can add their projects.

<h3>Creating your class</h3>	
	<p>Once you have successfully logged into your Teacher Account, if you are looking at the homepage, there will be a bar at the top of the screen with three options. Select “My Classes.”</p>
	<p>You can also access your classes from the dropdown under your username.</p>



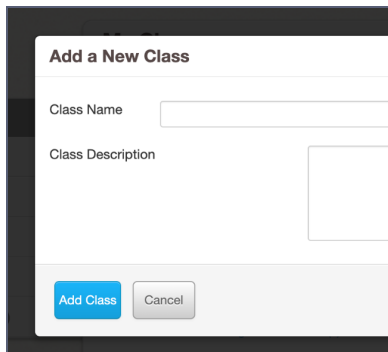


To create a class, click the “+ **New Class**” button at the top right of the page.

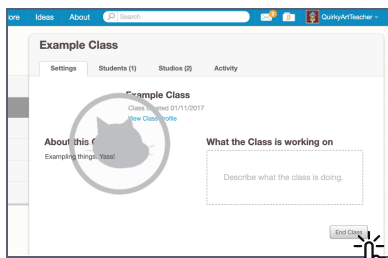
Enter the class name and description.

Warning: Do not include real names and locations, like the name of your school or city/town.

Once you have created a class, you can add students.



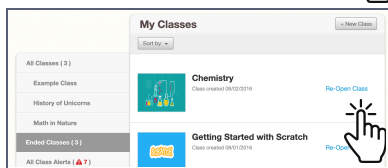
Ending your class



To end a class, under “My Classes,” choose your class and on the Settings tab, click the “End Class” button.

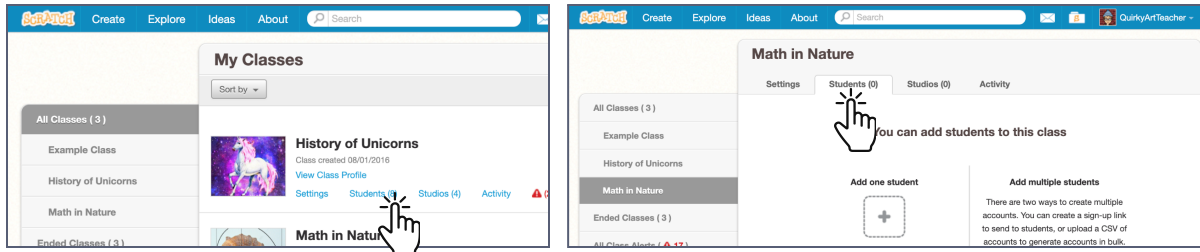
When you end a class, your class profile page will be hidden and your students will no longer be able to log in (but their projects and the class studios will still be visible on the site).

You may re-open the class at any time. By going to the “Ended Classes” tab and clicking the “Re-Open Class” link near the class you want to reopen.



Add Students to Your Class

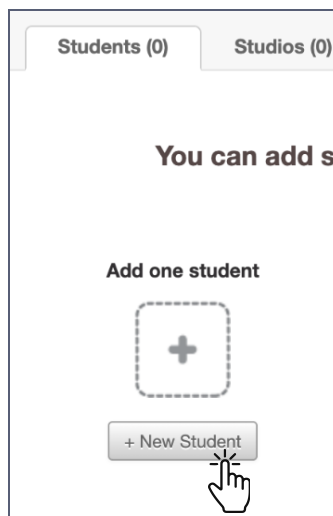
While on “My Classes,” select the class and then click on “Students” (either the link under the class name or the Students tab). Once created, your student accounts will appear here.



There are three ways to add students to your class. The first method allows you to add an individual student to a class. Methods 2 and 3 allow you to add multiple students to a class.

Tip: Create a naming convention as a guideline for generating usernames. For example, you may want each name to include an abbreviation for the course name, the class section, and the student’s number on your roster (ex: VisArts-02-17). Use the [Student Username List](#) we have created to record the usernames and passwords your students have created.

Method 1: Add Individual Students



Click the “+ New Student” button to add students individually.

Confirm the correct class is showing in the “Add to Class” dropdown menu.

You will be prompted to create a username for this student.

Warning: Make sure that the usernames you create do not contain identifying information about yourself, your students, or your school. Accounts that do not adhere to these guidelines will be deleted.

The password for this student username will automatically be set as the username of your teacher account.



Add New Student

Add to Class: Math in Nature

Username: type student username here

⚠️ Usernames must **not** reveal the identity of students in any way.

I understand that for safety and privacy, Scratch must **delete** any accounts which include real names, school name, or contact information.

Password: Will be set to the teacher's username: **QuirkyArtTeacher**

Add Student Cancel

Have students log into their accounts and change their passwords as soon as possible.

Tip: It is not possible to add an existing Scratch account to a classroom. You will need to create a new Student Account for them using your Teacher Account. A student can only be a part of one class, and it is not possible to transfer students from one class or teacher to another.

Method 2: Student Sign-up Link

Activity

Students to this class

Add multiple students

There are two ways to create multiple accounts. You can create a sign-up link to send to students, or upload a CSV of accounts to generate accounts in bulk.

Student Sign-up Link CSV Upload

Clicking the “Student Sign-Up Link” button brings you to another window and clicking the “Get Link” button will generate a link that will allow your students to join the class you have just created. The link will start with “http://scratch.mit.edu/signup...”

Students can then create their own usernames and passwords.

Warning: Remind your students that, when making their usernames, the username should not contain identifying information about themselves, their teacher, or their school. Accounts that do not adhere to these guidelines will be deleted.

Sign-up Link

Get a sign-up link that students can use to register for your class. They will be directed to a class page where they can “Join this Class”.

⚠️ Usernames must **not** reveal the identity of students in any way.

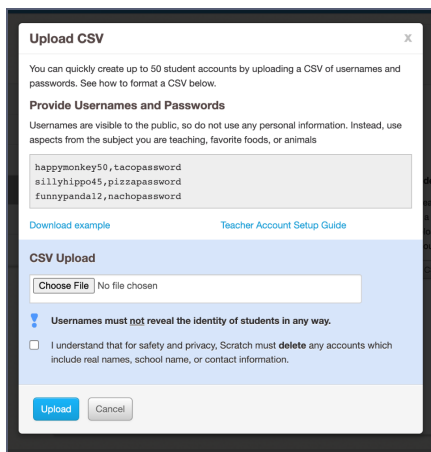
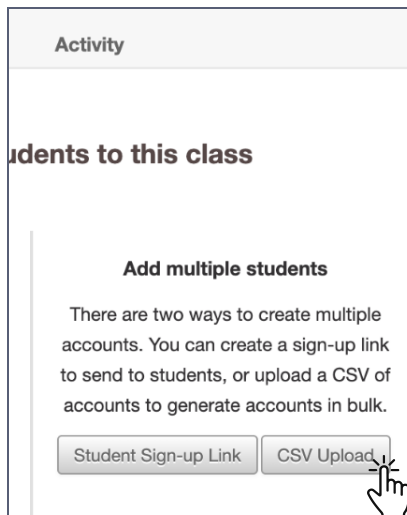
I agree to remind my students not to use their real names, school name, contact information or student ID numbers.

Get Link

Close



Method 3: CSV Upload



The screenshot shows a spreadsheet titled 'Student-Accounts-Template'. It has two columns, A and B, and two rows of student data.

	A	B
1	student1	password1
2	student2	password2

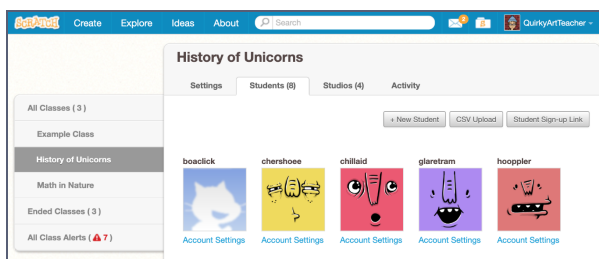
Click the “CSV Upload” button on the class page.

Using the template provided by clicking the “Download example” link, create a username and password for each of your students. You can use the template provided or create your own spreadsheet with student usernames in column A and passwords in column B. To upload your own template, you’ll need to save the file as a CSV file.

Once you’ve created usernames and passwords for each student and saved the file, click the “Choose file” button to locate the file, then click the “Upload” button.

It is not possible to add more than 250 students to a single class. You can, however, create a new class and add another 250 student accounts to each new class.

Warning: Make sure that the usernames you create do not contain identifying information about yourself, your students, or your school. Accounts that do not adhere to these guidelines will be deleted.



You can add students via any of these methods at any time under the “Students” tab.



Creating Studios for Student Work

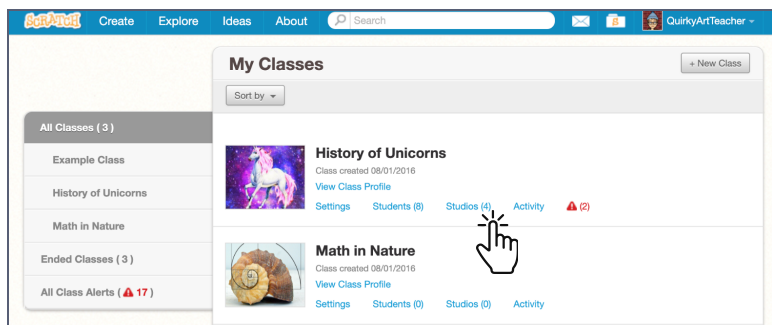
Studios allow you to create collections of student projects for specific classes or assignments. This makes it easier for you to view their projects throughout their creative process. It also makes it easier for students to collaborate and be inspired by each other's work.

Scratcher status is required in order to create a studio, and the person who created the studio is automatically assigned the role of "host." There is only one host per studio, and only studio hosts can edit the title, thumbnail, and description.

Studios are immediately public, even those created in the context of a class. Unlike Scratch projects, there is no share/unshare option for studios. Everyone can follow a studio, see studio comments and projects, and leave a comment or add a project (unless commenting or the ability to add projects is turned off).

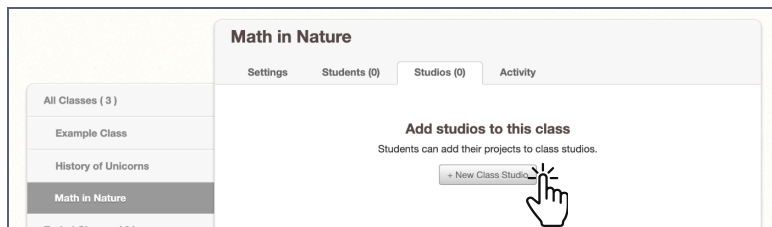
There are two ways to create a studio on a teacher account. Method one creates studios that automatically add all students in a class as curators. Method two creates studios without automatically adding students as curators, and students or any Scratcher can be individually added as curators.

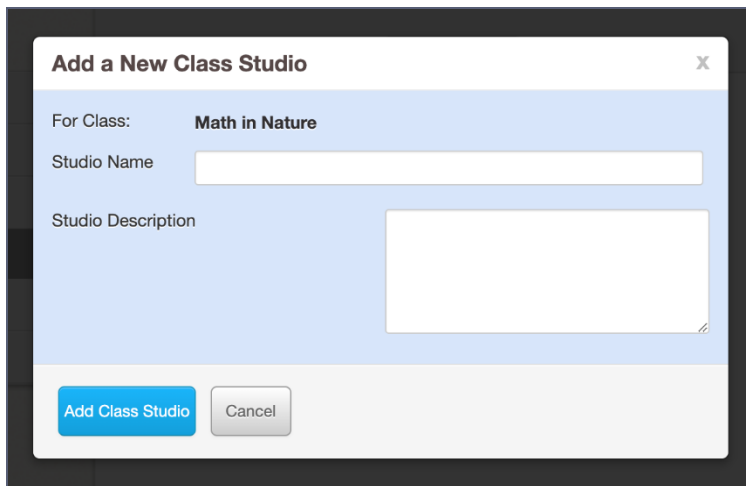
Method 1: Create a studio that automatically adds all students in a class as curators



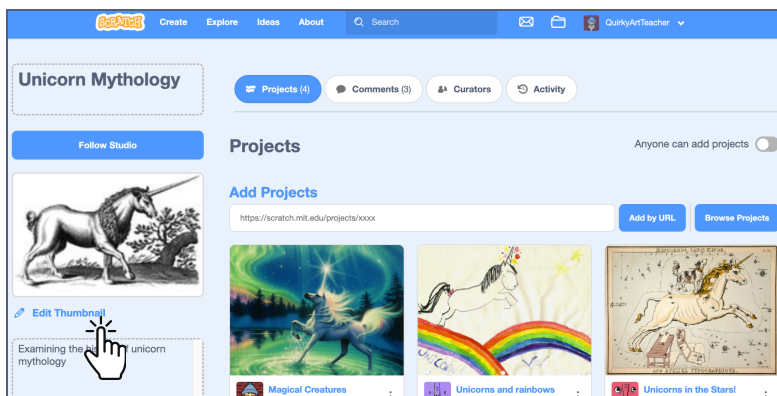
Once logged into your Scratch account, go to "My Classes."

Choose the class to assign the studio to, then click on "Studios" (either the link under the class name or the Studios tab). Then click the "+ New Class Studio" button.





On the window that appears, you will be asked to **give the studio a name and description**. (These can always be adjusted in the studio later.) In the description, you can share the theme of the studio, what kinds of projects you are looking to include... Just be sure your title and description don't reveal any personal information (like school name or first and last name).

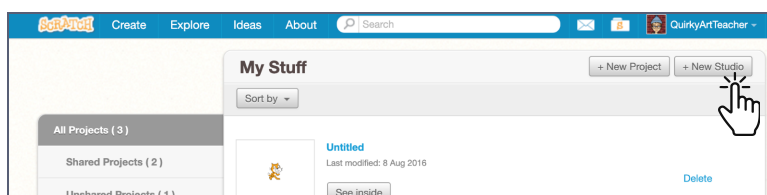


Then, click the "Add Class Studio" button.

Once in the studio, click the "Edit Thumbnail" button to change the default gray cat image in the upper left-hand corner. **Upload your own studio thumbnail image**. The maximum file size for a thumbnail is 512 KB and your image must be less than 500x500 pixels.

When you click on the "Curators" tab, you should see all the class students have been set as studio curators.

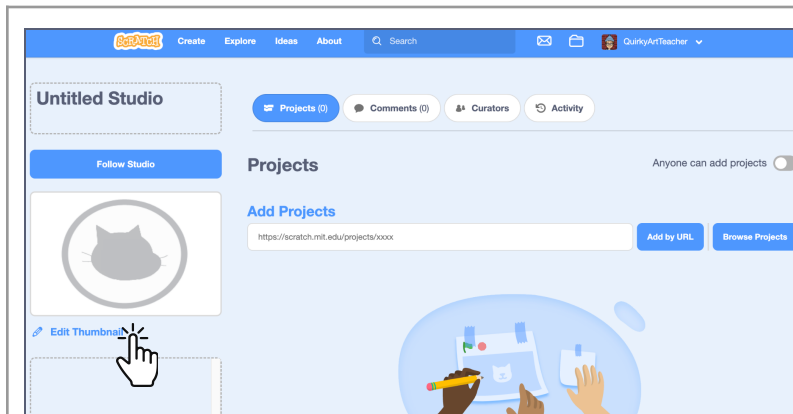
Method 2: Create a studio without automatically adding students as curators



Once logged into your Scratch account, go to "My Stuff."

Choose the "+ **New Studio**" button at the top right.





Click on “Untitled Studio” to **give your studio a name and description**. In the description, you can share the theme of the studio, what kinds of projects you are looking to include... Just be sure your title and description don’t reveal any personal information (like school name or first and last name).

Click the “Edit Thumbnail” button to change the default gray cat image in the upper left-hand corner. **Upload your own studio thumbnail image**. The maximum file size for a thumbnail is 512 KB and your image must be less than 500x500 pixels.

When you click on the “Curators” tab, you should see no curators have been assigned yet.

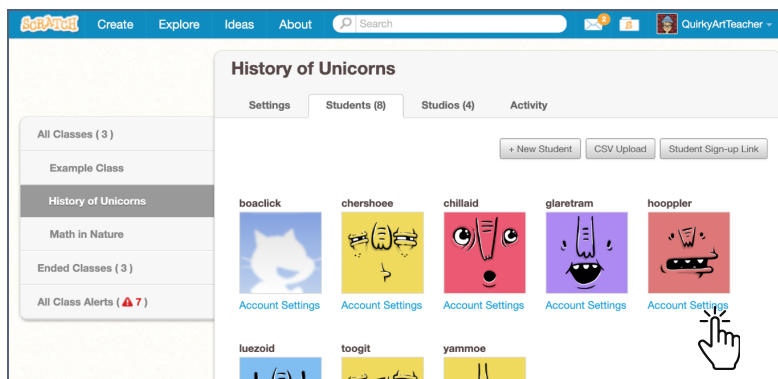
See our [Studio Guide](#) for detailed information on:

- Studio Definitions
- How to Manage a Studio
- How to Add Projects to a Studio

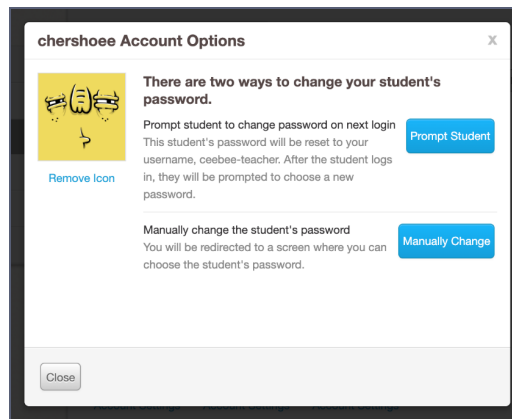


Managing Your Students

Managing a student



You can manually **reset a student password** from within your Scratch Teacher Account. First, navigate to “My Classes” and choose the class and go to the “Students” tab. Then click on the “Account Settings” link below the student’s account.



You cannot delete a student’s account by using a Teacher Account, but students can delete their own account.



You can see alerts about notifications your students receive on the “Activity” tab of a class or the “All Class Alerts” tab.

Tip: If you’d like to translate this guide, [click here to make a copy](#) of this Google doc.

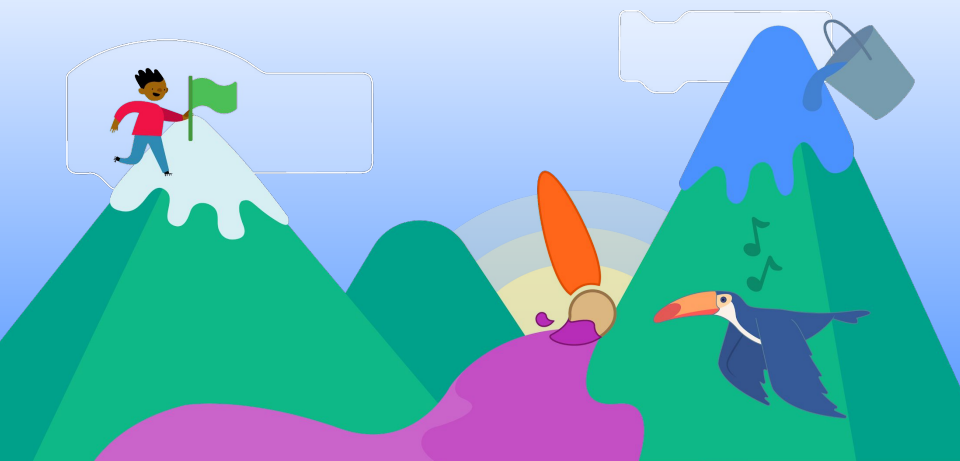


Getting Started with

SCRATCH

Beginner's Guide

Create your own games, animations,
interactive stories, and more.



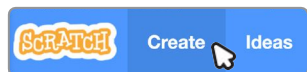


GETTING STARTED

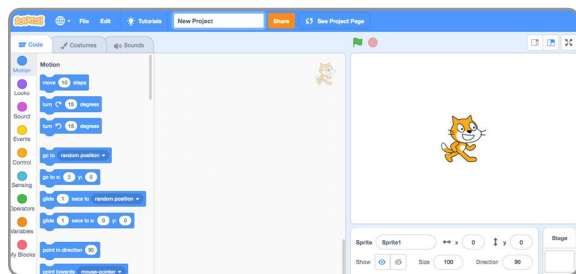
You can use Scratch online at: scratch.mit.edu

scratch.mit.edu

Once you've navigated to scratch.mit.edu, click **Create**.



This will bring you to the **Scratch Editor**, where you can start creating projects.



If your computer uses an older operating system, or your internet connection is unreliable, you can download Scratch and use it offline.



Visit: <https://scratch.mit.edu/download>
for information on downloading and installing the Scratch app.



THE SCRATCH EDITOR

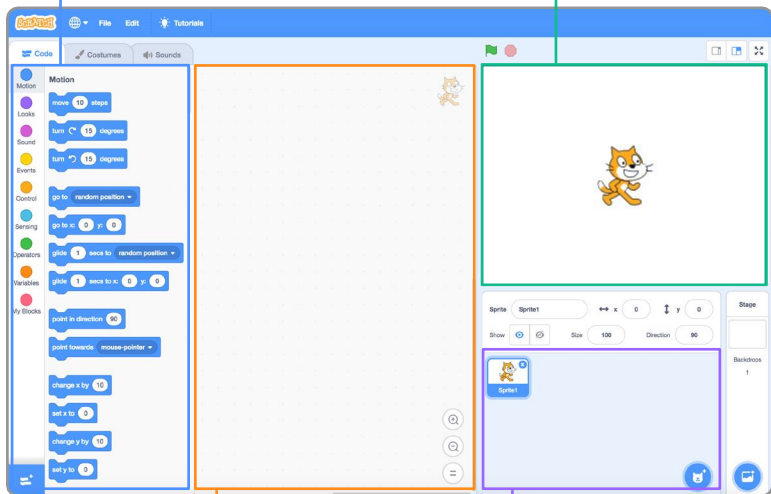
The Scratch Editor is where you create projects in Scratch. Here are its main parts:

Blocks Palette

Blocks for coding your projects

The Stage

Where your creations come to life



Coding Area

Drag in blocks and snap them together to code your sprites

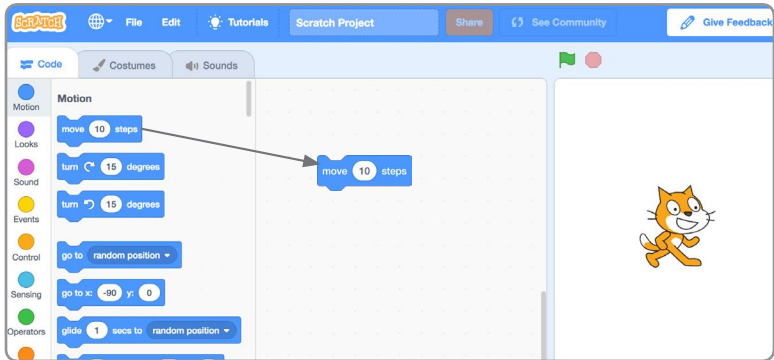
Sprite List

Click the thumbnail of a sprite to select it

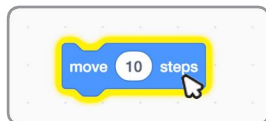


LET'S CODE!

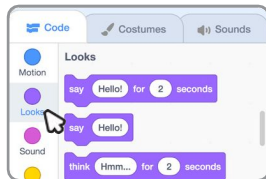
To code projects in Scratch, you snap together blocks. Start by dragging out a **move** block.



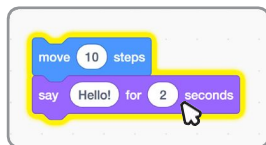
Click the block to try it.
Does your cat move?



Now say something!
Click the **Looks** category.



Drag out a **say** block.
Snap it onto the **move** block. Click on your blocks to try them.





WHAT IS A SPRITE?

In Scratch, any character or object is called a sprite. Every new project in Scratch starts with the Cat sprite.



Want to choose a different sprite?

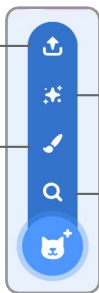


Click the New Sprite icon.

Or, hover over the **New Sprite** icon to see more options.

Upload an image from your computer.

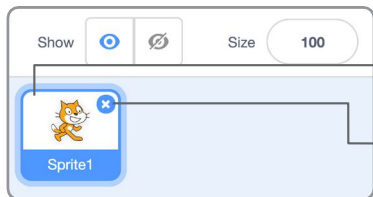
Draw your own sprite.



Click for a surprise sprite!

Choose a sprite from the library.

Want to **delete a sprite** from your project?



First, select the sprite by clicking on its thumbnail in the Sprite List.

Then, click here to delete the sprite.



WHERE IS YOUR SPRITE?

Every sprite has an **x** and **y** position on the Stage.

x is the position of the sprite from left-to-right.

y is the position from top-to-bottom.

At the very center of the stage, **x** is 0 and **y** is 0.



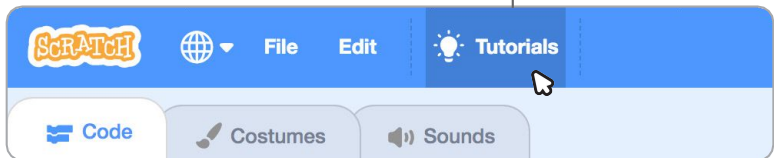
When you move your sprite, you can see its **x** and **y** position change.



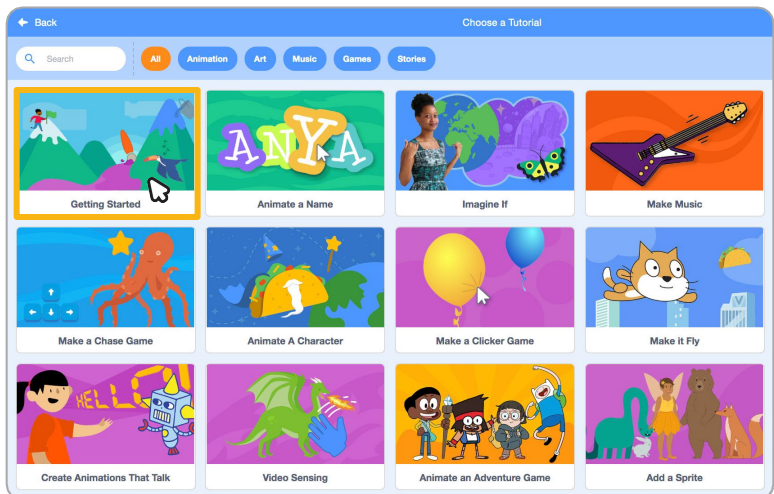
TUTORIALS

There are a range of tutorials available in the Scratch **Tutorials Library**, which guide learners in creating projects with Scratch. Students can get started making their own stories, animations, and games.

You can get to the Tutorials Library from the Scratch Editor by clicking the **Tutorials** button.



The **Getting Started** tutorial will walk you through the basics.

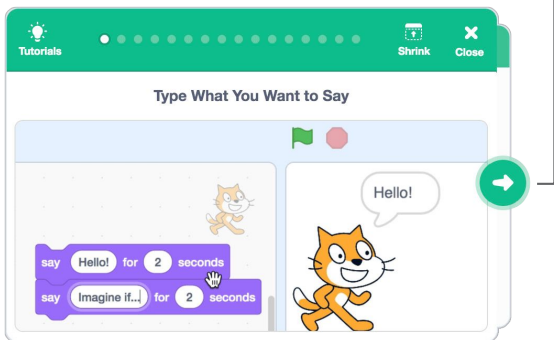




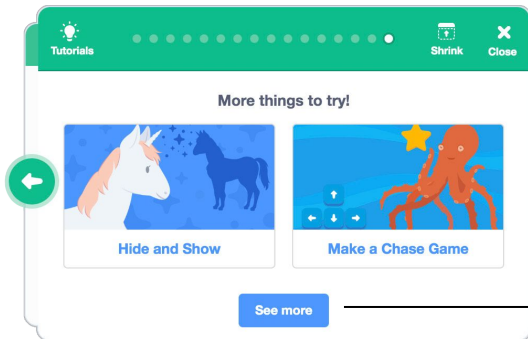
TUTORIALS

Once you've selected the tutorial, it will open in the Scratch Editor.

Click the green arrow to see each step.



When you've reached the end of a tutorial you can select another tutorial, and keep adding to your project.



Click here to see all the Tutorials.

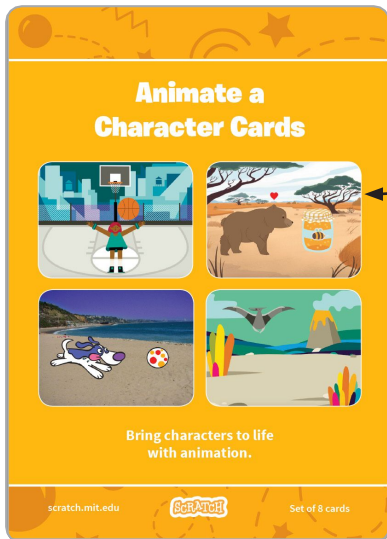


CODING CARDS

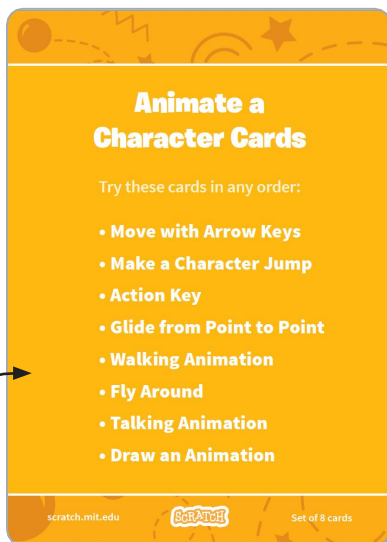
The Scratch **Coding Cards** provide another way to learn to create projects with Scratch. Download the cards at scratch.mit.edu/ideas.

Each set of cards starts with a title card, which shows you what you can create.

The **Animate a Character** cards are a great set to start with.



Examples of what you can create



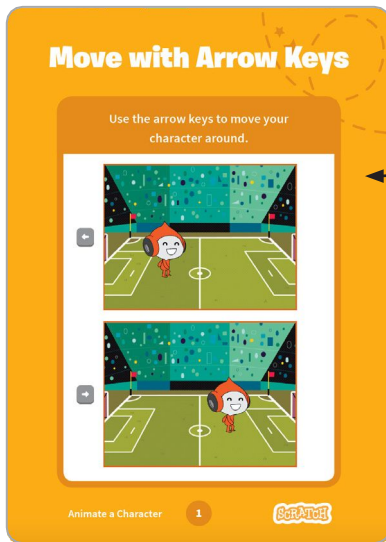
A list of all the cards in this set



USING THE CODING CARDS

After each title card is a series of cards walking you through each step of creating a project.

Add your own sprites, backdrops and more!



The front of each card shows you what you can create.



The back shows you how to do it.



GET CREATIVE!

Encourage students to use their imagination as you create projects. There are many different ways they can make their Scratch projects unique.

You can choose or draw your own characters.



Choose a sound or record your own.



Try changing numbers or adding blocks to your code to see what happens.



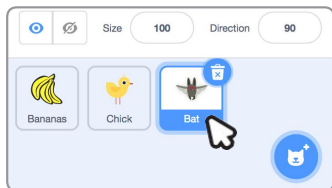
Experiment and customize your project however you want!



GET CREATIVE WITH SPRITES!

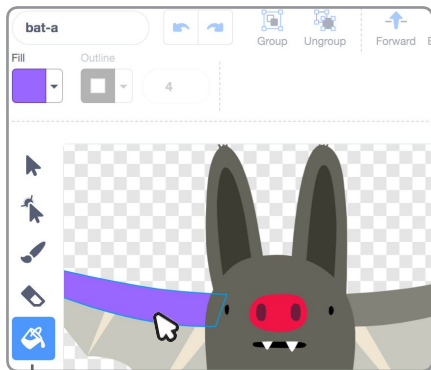
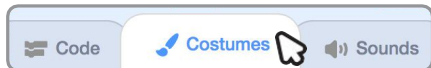
Scratch has its own paint tools, which allow you to customize sprites from the library, or even create sprites of your own.

Let's start by editing a sprite from the library.



Select a sprite to edit by clicking on it in the Sprite list.

Click the **Costumes** tab at the top left to see the paint tools.



The paint tools allow you to recolor sprites, add to them with a paint brush, and change them in a variety of ways.

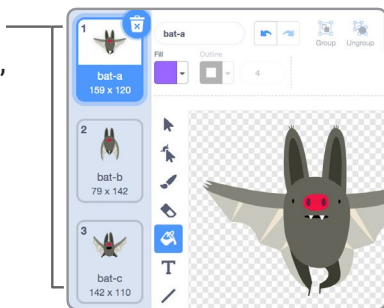
You can use the **paint bucket** tool to recolor different parts of a sprite.



GET CREATIVE WITH SPRITES!

Some sprites, like the Bat sprite have multiple costumes, or poses.

You can see a sprite's costumes by clicking the **Costumes** tab.

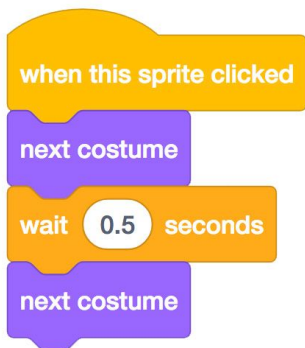


If your sprite only has one costume, right click on the costume to duplicate it (On Mac control + click).



Now you can modify the second costume using the paint tools, so your sprite has two different poses or facial expressions.

Click the **Code** tab, then try adding these blocks.

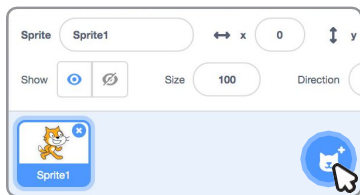




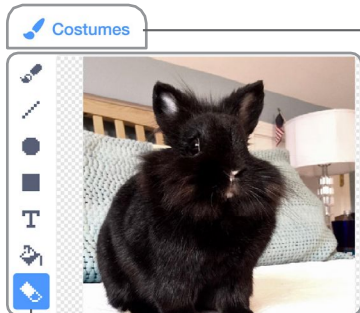
ADD YOUR OWN PHOTOS

There are many ways to create your own sprites and artwork using the Scratch paint tools.

You can create your own sprites by uploading photos or images and erasing the background.

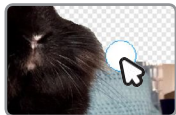


Hover over the New Sprite button, then select **Upload Sprite**.



Next click the **Costumes** tab. You will see bitmap tools for editing your image.

Click the **eraser** icon and use the eraser tool to remove the background from your photo.



Tip: to adjust the size of the eraser, type a larger or smaller number.

There are two modes for drawing in Scratch:

1. **Bitmap Mode** allows you to edit photos and paint with pixels.
2. **Vector Mode** allows you to create and edit shapes.

Tip: If you'd like to remix and customize this guide, [click here to make your own copy](#) of the Google Slides template.



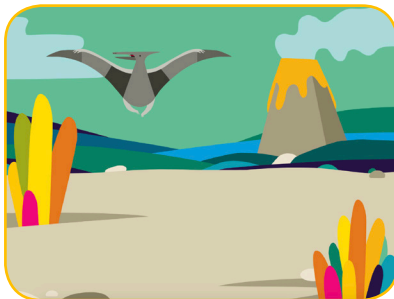
Created by the Scratch Team (scratch.mit.edu) and shared under the Creative Commons Attribution-ShareAlike 4.0 International Public License (CCbySA 4.0).

||| ||| ||| ||| ||| COLOR SCHEME

EDUCATOR GUIDE

Animate a Character

With this guide, you can plan and lead a 55-minute lesson using Scratch. Students will gain experience with coding as they bring characters to life with animation.



Lesson Outline

Objective: Students will become familiar with the Scratch environment by animating a character.



IMAGINE
10 minutes

First, gather as a group to introduce the theme and spark ideas.



CREATE
40 minutes

Next, help students as they animate characters, working at their own pace through the tutorial.



SHARE
5 minutes

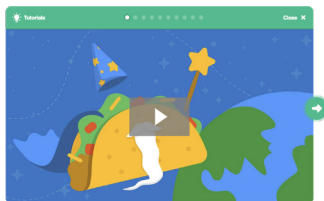
At the end of the session, gather together to share and reflect.

Get Ready for the Lesson

Use this checklist to prepare for the lesson.

Preview the Tutorial

The *Animate a Character* tutorial shows students how to create their own projects. Preview the tutorial before your lesson and try the first few steps: scratch.mit.edu/tutorials



Print the Activity Cards (optional)

Print a few sets of *Animate a Character* cards to have available for students during the lesson. scratch.mit.edu/ideas



Make sure students sign into their Scratch accounts

Have students sign into their own Scratch accounts at scratch.mit.edu.

Set up computers or laptops

Arrange computers so that students can work individually or in pairs.

Set up a computer with projector or large monitor

You can use a projector to show examples and demonstrate how to get started.

Imagine



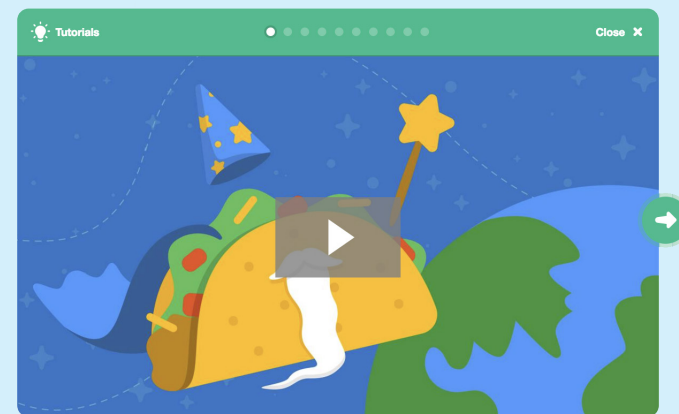
Begin by gathering the students to introduce the theme and spark ideas for projects.

Warm-up Activity: Favorite Characters

Gather the group in a circle. Ask each student to say their name, then share a favorite character from a book, movie, or TV show, and one or two of their favorite things about that character.

Provide Ideas and Inspiration

To spark ideas, watch the *Animate a Character* video at the start of the tutorial. The video shows a variety of projects to spark ideas and inspiration.



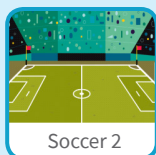
View the scratch.mit.edu/ideas

Demonstrate the First Steps



Demonstrate the first few steps of the tutorial so students can see how to get started.

Choose a backdrop.



Choose a character to animate.



Make your sprite move right and left with arrow keys:

when right arrow key pressed
change x by 10

Choose right arrow from the menu.

when left arrow key pressed
change x by -10

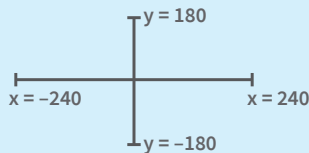
Choose left arrow from the menu.

Type a minus sign to move left.

Press the left arrow and right arrow keys on your keyboard to move.



Helpful Hint: Understanding x y coordinates will help students figure out how to move sprites around the stage.



y is the position on the Stage from top to bottom.

x is the position on the Stage from right to left.

Create



Support students as they create animated Scratch projects.

Start with Prompts

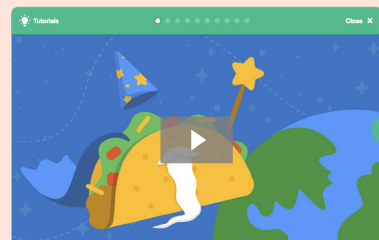
Ask students questions to get started

Which character would you like to animate?

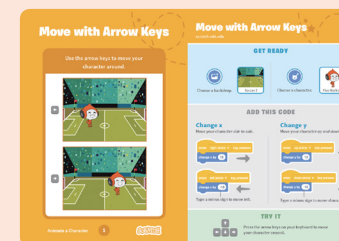
What do you want your character to do?

Provide Resources

Offer options for getting started



Some students may want to follow the online tutorial: scratch.mit.edu/animate



Others may want to explore using the activity cards: scratch.mit.edu/ideas

Suggest Ideas for Starting

- Choose a character to animate.
- Animate your character: make it jump, fly, glide or talk!
- Choose a backdrop.



CREATE

More Things to Try

- Try combining more than one kind of animation.
- If you're not sure what to do, pick a card and try something new.
- Add a second character or object to animate.



Support collaboration

- When someone gets stuck, connect them to another participant who can help.
- See a cool idea? Ask the creator to share with others.



Encourage experimenting

The Animate a Character activity can be done in any order, with a range of different character and object sprites.

Encourage students to try new things:

What will your character do next?

How can you make your animation interactive?



SHARE

Share

Have students share their project with their neighbors.

Ask questions they can discuss:

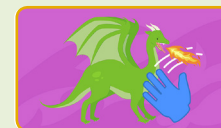
What do you like best about the project you made?

What was the hardest part?

If you had more time, what would you add or change?

What's Next?

Students can use the ideas and concepts from this lesson to create a wide variety of projects. Encourage them to continue developing their projects into games, stories or interactive art with the resource listed below.



Video Sensing

Interact with characters and objects in Scratch with video sensing.

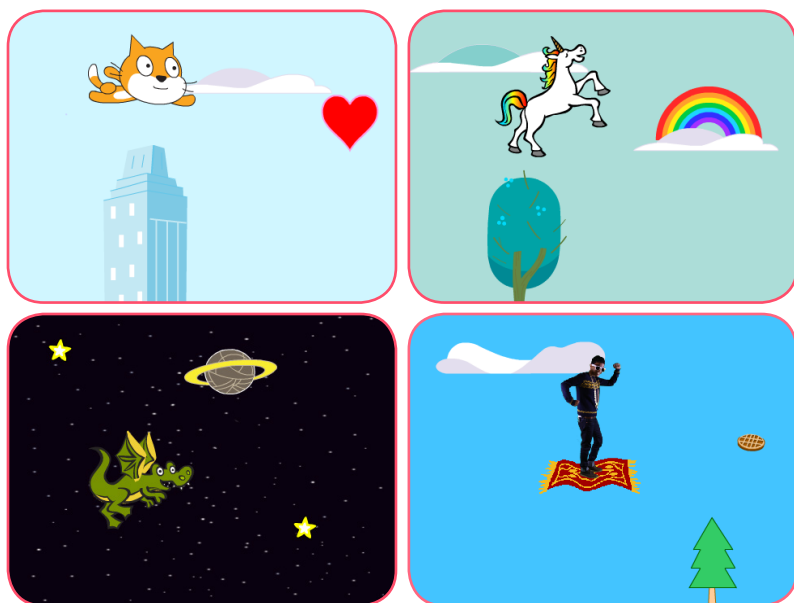
Find this project and more in the Tutorials library: scratch.mit.edu/ideas

Scratch is a project of the Lifelong Kindergarten Group at the MIT Media Lab.

EDUCATOR GUIDE

Make It Fly

With this guide, you can plan and lead a 55-minute lesson using Scratch. Students will choose a character and program it to fly.



Lesson Outline

Objective: Students will create an animation with the illusion of a flying character.



IMAGINE
10 minutes

First, gather as a group to introduce the theme and spark ideas.



CREATE
40 minutes

Next, help students as they create a flying animation, working at their own pace through the tutorial.



SHARE
5 minutes

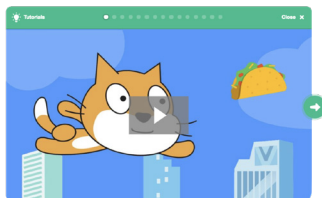
At the end of the session, gather together to share and reflect.

Get Ready for the Lesson

Use this checklist to prepare for the lesson.

Preview the Tutorial

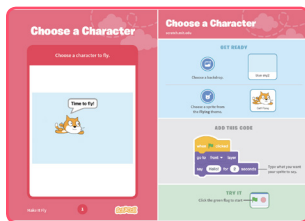
The *Make It Fly* tutorial shows students how to create their own projects. Preview the tutorial before your lesson and try the first few steps: scratch.mit.edu/fly



Print the Activity Cards (optional)

Print a few sets of *Make It Fly* cards to have available for students during the lesson.

scratch.mit.edu/fly/cards



Make sure students sign into their Scratch accounts

Have students sign into their own Scratch accounts at scratch.mit.edu.

Set up computers or laptops

Arrange computers so that students can work individually or in pairs.

Set up a computer with projector or large monitor

You can use a projector to show examples and demonstrate how to get started.

Imagine



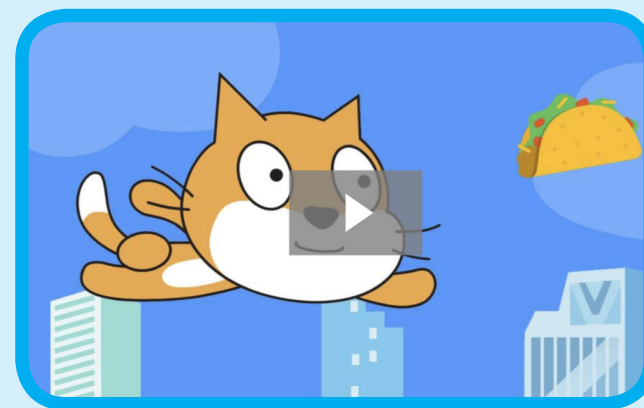
Begin by gathering the students to introduce the theme and spark ideas for projects.

Warm-up Activity: If I Could Fly...

Gather the group in a circle and ask, “If you could fly, where would you want to go?” Suggest that they close their eyes and imagine flying through their favorite place. Ask, “Where are you? What kinds of things do you see below you?” If there’s time, have each person say where they imagined flying or something they saw on their flight.

Provide Ideas and Inspiration

Show the introductory video for the *Make It Fly* tutorial. The video shows a variety of projects for ideas and inspiration.



View at scratch.mit.edu/fly or vimeo.com/llk/fly

Demonstrate the First Steps



Demonstrate the first few steps of the tutorial so students can see how to get started.

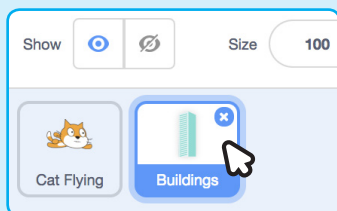
In Scratch, click Create.
Choose a flying sprite from the library:



Choose a new sprite for your character to fly past:



Make the building move across the stage to make your character look like it's flying:



Create



Support students as they make a flying animation.

Start with Prompts

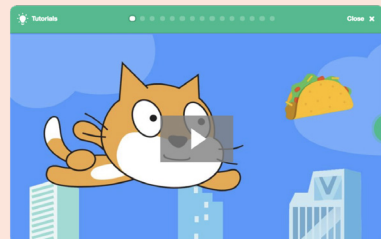
Ask students questions to get started

What character would you like to make fly?

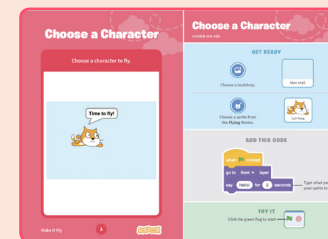
Where will your character go flying?

Provide Resources

Offer options for getting started



Some students may want to follow the online tutorial:
scratch.mit.edu/fly



Others may want to explore using the activity cards:
scratch.mit.edu/fly/cards

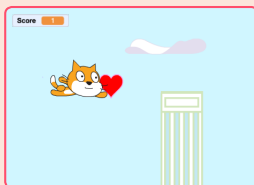
Suggest Ideas for Starting

- Choose a character
- Choose buildings or other scenery
- Make the character say something
- Make the scenery move



More Things to Try

- Switch costumes to change the scenery.
- Make your character move when you press a key.
- Add clouds and other floating objects.
- Score points when touching an object.



Encourage Debugging

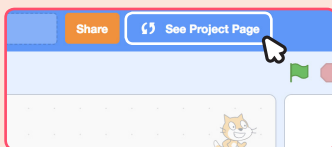
Here are some strategies to suggest to help students fix any bugs or difficulties they encounter:

- When stuck, talk out what you're working on with someone.
- Try out small bits of code at a time to figure out what's happening at each step.
- Look closely at the blocks on the tutorial or activity cards to see if they are the same or different from the blocks you're using.
- Remember that bugs always arise when creating a computer program. Debugging is a helpful skill to know not just in coding, but throughout life.

Prepare to Share

To add instructions and credits to a project, click the button: "See project page".

Give your project a title, add instructions and credits, then click Share.



Share



Share

Share projects with others in the room. Organize a flying character showcase. Ask half the room show their projects, while the others view them. Then switch.

Suggest that they ask each other questions, such as:

What do you like best about the project you made?

What might you like to change or make next?

What's Next?

Students can use the ideas and concepts from this lesson to create other projects. Here are a couple of variations on the flying character project you could suggest.



Flying Game

Make a game where you avoid some objects and try to catch others. Add or subtract points based on what your character touches.



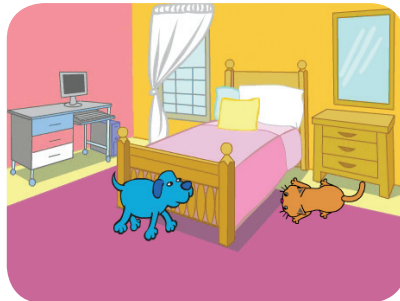
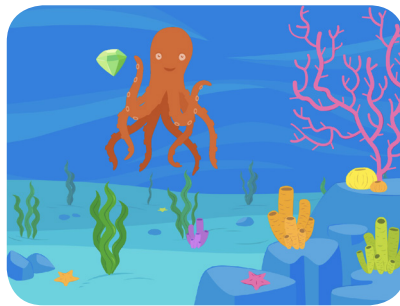
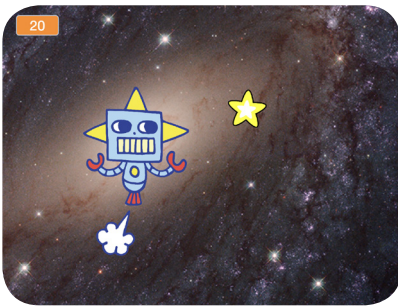
Flying Stories

Tell a story about your flying characters. You can record your voice and play sound clips. Or, use say blocks to make voice bubbles.

EDUCATOR GUIDE

Make a Chase Game

With this guide, you can plan and lead a 55-minute lesson using Scratch. Students will make a game that includes a variable to keep score.



Lesson Outline

Objective: Students will create a game using sensing.



IMAGINE
10 minutes

First, gather as a group to introduce the theme and spark ideas.



CREATE
40 minutes

Next, help students as they make chase games, working at their own pace through the tutorial.



SHARE
5 minutes

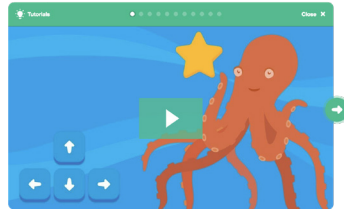
At the end of the session, gather together to share and reflect.

Get Ready for the Lesson

Use this checklist to prepare for the lesson.

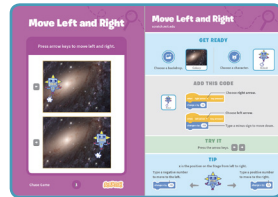
Preview the Tutorial

The *Make a Chase Game* tutorial shows students how to create their own projects. Preview the tutorial before your lesson and try the first few steps,



Print the Activity Cards (optional)

Print a few sets of *Chase Game* cards to have available for students during the lesson. You can download the cards at: scratch.mit.edu/ideas



Make sure students sign into their Scratch accounts

Have students sign into their own Scratch accounts at scratch.mit.edu.

Set up computers or laptops

Arrange computers so that students can work individually or in pairs.

Set up a computer with projector or large monitor

You can use a projector to show examples and demonstrate how to get started.

Imagine



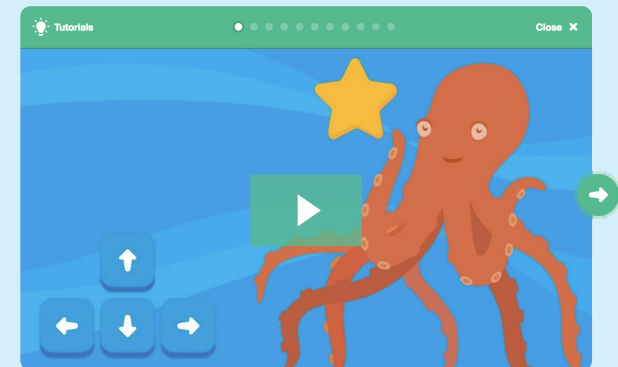
Begin by gathering the students to introduce the theme and spark ideas for projects.

Warm-up Activity: Imaginary Chase

Gather the students in a circle. Start by giving an example of one thing chasing another, such as “The dog is chasing the dinosaur.” The next person adds on, such as, “The dinosaur is chasing a donut.” The following person adds on by saying, “The donut is chasing a duck.” or whatever creature or object they choose. Continue until each person has added on to this imaginary game of chase.

Provide Ideas and Inspiration

To spark ideas, watch the *Make a Chase Game* video at the start of the tutorial.



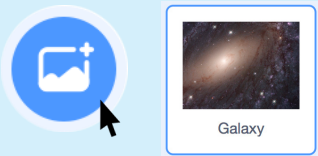
View the video at scratch.mit.edu/chase

Demonstrate the First Steps

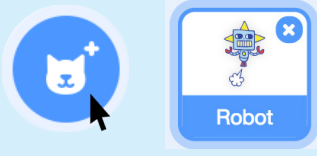


Demonstrate the first few steps of the tutorial so students can see how to get started.

Choose a backdrop.



Choose a Sprite, like Robot.



Make your sprite move right and left with arrow keys.

when **right arrow** key pressed

change x by **10**

Choose right arrow from the menu.



when **left arrow** key pressed

change x by **-10**





Choose left arrow from the menu.

Type a minus sign to move left.

Press the left arrow and right arrow keys on your keyboard to move.

Discuss next steps they can try, such as coding the sprite to move up and down and adding a sprite to chase.

Create



Support students as they create catch games. Suggest working in pairs.

Start with Prompts

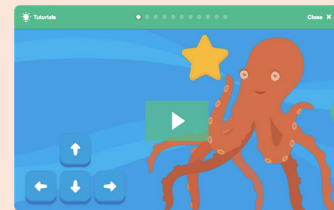
Ask students questions to get started

Which backdrop would you like to choose for your game?

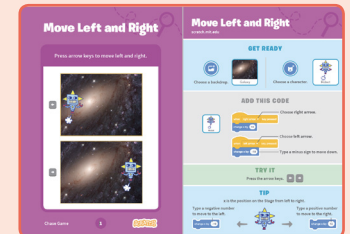
Who do you want as the main character in your game? What will it chase?

Provide Resources

Offer options for getting started



Some students may want to follow the online tutorial: scratch.mit.edu/chase



Others may want to explore using the printed cards: scratch.mit.edu/ideas

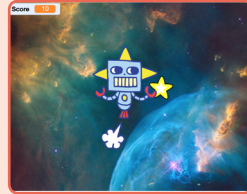
Suggest Ideas for Starting

- Choose a backdrop
- Choose or draw a main character
- Make it move with arrow keys.
- Select an object to chase.



More Things to Try

- Code the star or other sprite to chase
- Add a variable to keep score
- Add sounds
- Add a level
- Show a message when reaching the new level



Encourage Tinkering

- Encourage students to feel comfortable trying combinations of blocks and seeing what happens.
- Suggest students look inside other chase games to see the code.
- If they find code they like, they can drag the scripts or sprites into the backpack to reuse in their own project.

Prepare to Share

To add instructions and credits to a project, click the button: “*See project page*”.



Share

Have students share their projects with their neighbors.

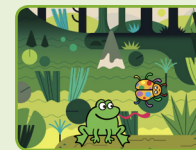
Ask questions that encourage reflection:

What do you like best about your game?

If you had more time, what would you add or change?

What's Next?

Chase Game projects provide an introduction to creating interactive games in Scratch. Here are a few ways that learners can build on the concepts they learned from this project.



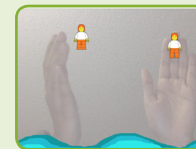
Add Obstacles

For a more complex game, add obstacles to avoid. Subtract points when you hit the obstacles.



Make a Two-Player Game

For an extra challenge, make a version of the game that allows two players to play.



Video Sensing

If the computers have a web camera attached or built-in, learners can make a game that they interact by moving their bodies. See the Video Sensing tutorial and educator guide for support.

Created by the Scratch Team

EDUCATOR GUIDE

Pong Game

With this guide, you can plan and lead a 55-minute lesson using Scratch. Students will gain experience with coding as they design a bouncing ball game.



Lesson Outline

Objective: Students will develop an interactive game using variables to keep score.



IMAGINE
10 minutes

First, gather as a group to introduce the theme and spark ideas.



CREATE
40 minutes

Next, help students as they make games, working at their own pace through the tutorial.



SHARE
5 minutes

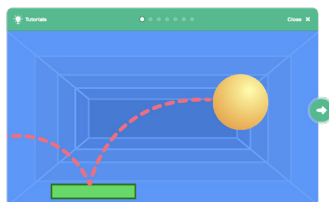
At the end of the session, gather together to share and reflect.

Get Ready for the Lesson

Use this checklist to prepare for the lesson.

Preview the Tutorial

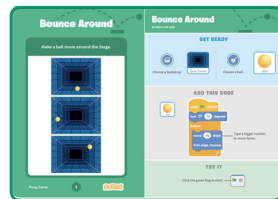
The *Pong Game* tutorial shows students how to create their own projects. Preview the tutorial before your lesson and try the first few steps: scratch.mit.edu/pong



Print the Activity Cards (optional)

Print a few sets of *Pong Game* cards to have available for students during the lesson.

scratch.mit.edu/ideas



Make sure students sign into their Scratch accounts

Have students sign into their own Scratch accounts at scratch.mit.edu.

Set up computers or laptops

Arrange computers so that students can work individually or in pairs.

Set up a computer with projector or large monitor

You can use a projector to show examples and demonstrate how to get started.

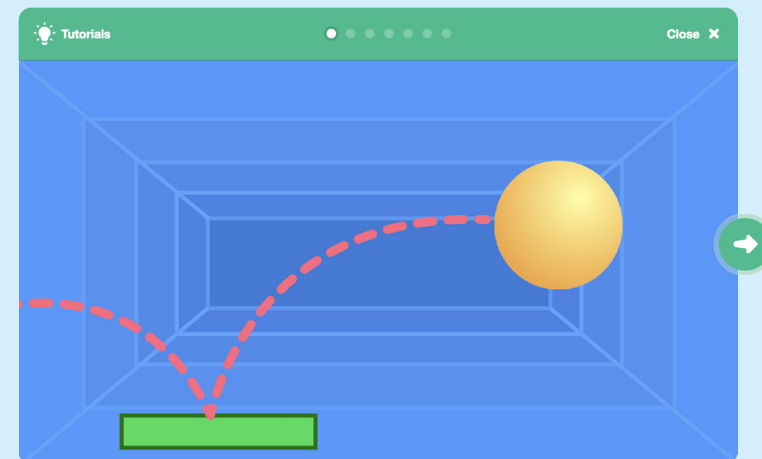
Imagine



Begin by gathering the students to introduce the theme and spark ideas for projects.

Provide Ideas and Inspiration

Show the introductory video for the *Pong Game* tutorial. The video shows pong games with a variety of themes, including everything from soccer to a magic potion-themed Pong game.



View at scratch.mit.edu/pong

Warm-up Activity: Bouncing Ideas

To get students thinking about a theme for their game, take turns calling out a theme, such as pizza pong or flower pong and brainstorming ideas for the type of images they could use to represent the theme.

Demonstrate the First Steps



Demonstrate the first few steps of the tutorial so students can see how to get started.

Go to the Scratch website. Click Create. Choose a new backdrop:



Choose a ball sprite and make it bounce around:



Add a paddle sprite and control it with the mouse:



Create



Support students as they create pong games, on their own or in pairs.

Start with Prompts

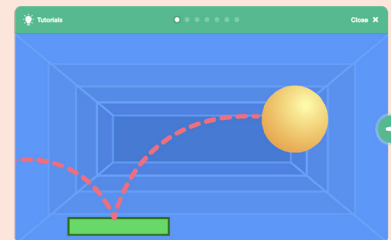
Ask students questions to get started

What background do you want for your game?

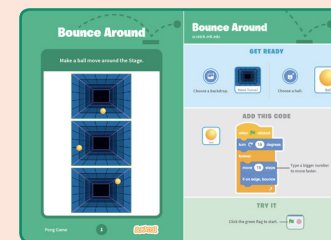
What color or type of ball?

Provide Resources

Offer options for getting started



Some participants may want to follow the online tutorial:
scratch.mit.edu/pong



Others may want to use the printed activity cards:
scratch.mit.edu/ideas

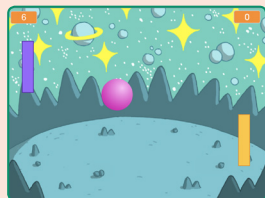
Suggest Ideas for Starting

- Choose a backdrop
- Choose or draw a ball sprite and make it bounce around
- Add a paddle sprite that you can control
- Make the ball bounce off the paddle



More Things to Try

- Add sounds and color effects
- Keep score by adding a variable
- Add a way to win or lose the game
- Change the backdrop when you reach a certain number of points
- Duplicate the ball for an added challenge



Offer strategies for problem solving

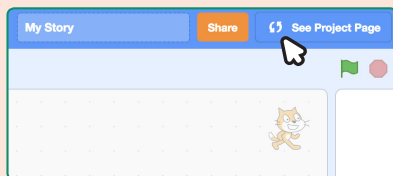
- Talk out what you're working on with someone
- Try out small bits of code at a time to figure out what's happening at each step
- Look closely at the blocks on the tutorial or activity cards to see if they are the same or different from the blocks you're using
- Look at the code for other pong games on the Scratch site



Prepare to Share

To add instructions and credits to a project, click the button: "See project page".

Then click the Share button if you want the project visible to others online.



Share

Have participants share their projects with others in the room.

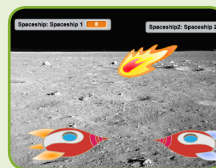
Ask questions to encourage reflection:

What did you notice about the games you tried?

What ideas might you add to your game?

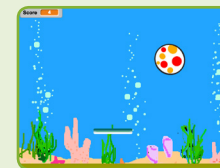
What's Next?

Here are a couple of other directions you could suggest:



Two-Player Game

For a more advanced project, try making a two-player game. To make a new version of your own project, click **File > Save as a Copy**.



Remix a Game

A different way to make a pong game is to remix someone else's project, adding images and ideas. Find a project to remix in the **Pong Game Studio**: scratch.mit.edu/studios/644508/ Click '**See inside**', then click the '**Remix**' button.

Scratch is a project of the Lifelong Kindergarten Group at the MIT Media Lab.

EDUCATOR GUIDE

Create a Story

With this guide, you can plan and lead a 55-minute lesson using Scratch. Students will create a story with settings, characters, and dialogue.



Lesson Outline

Objective: Students will create an animated story between at least two characters.



IMAGINE
10 minutes

First, gather as a group to introduce the theme and spark ideas.



CREATE
40 minutes

Next, help students as they create story projects, working at their own pace through the tutorial.



SHARE
5 minutes

At the end of the session, gather together to share and reflect.

Get Ready for the Lesson

Use this checklist to prepare for the lesson.

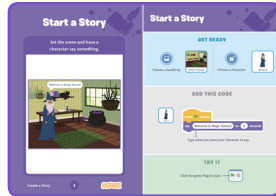
Preview the Tutorial

The *Create a Story* tutorial shows students how to create their own projects. Preview the tutorial before your lesson and try the first few steps: scratch.mit.edu/story



Print the Coding Cards (optional)

Print a few sets of *Create a Story* cards to have available for students during the lesson. You can download from this page: scratch.mit.edu/ideas



Make sure students sign into their Scratch accounts

Have students sign into their own Scratch accounts at scratch.mit.edu.

Set up a studio for project sharing on Scratch

Set up a studio so students will be able to add their projects. Go to your *My Stuff* page, then click the **+New Studio** button. Type in a name for the studio.

Set up computers or laptops

Arrange computers so that students can work individually or in pairs.

Imagine



Begin by gathering the students to introduce the theme and spark ideas for projects.

Warm-up Activity: Story Starters in a Bag

Have students make up a brief story by giving them a bag with three objects in it, and asking them to include all of the items in the story. In each bag, you could include small objects, pictures of animals or characters, and/or words (people, places, or things). Divide students into groups of two or three, and have each pick a bag. Give them a few minutes to come up with a quick story.

Provide Ideas and Inspiration

You can show the *Create a Story* tutorial video to show students how they can start making stories in Scratch.



View the video at: scratch.mit.edu/story

Demonstrate the First Steps



Demonstrate the first few steps of the tutorial so students can see how to get started.

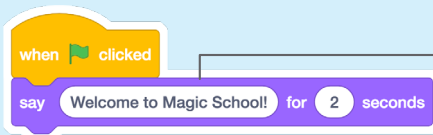
In Scratch, click Create.
Choose a backdrop.



Choose any character (in Scratch called a *sprite*).



Code your character to say something.



Type what you want your character to say.

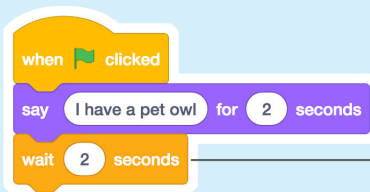
Click the green flag to start.



Add another character.



Add code to the new character.



Use this block to have the second character wait before they say something.

Create



Support students as they create Story projects, on their own or in pairs.

Start with Prompts

Ask students questions to get started

Where will your story take place?

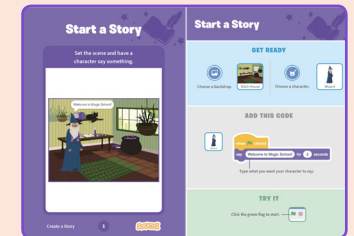
What will happen first?

Provide Resources

Offer options for getting started



Some students may want to follow the online tutorial: scratch.mit.edu/story



Others may want to explore using the coding cards: scratch.mit.edu/ideas

Suggest Ideas for Starting

- Choose a backdrop.
- Choose a character.
- Make a character say something
- Make a character hide and show.



More Things to Try

- Switch backdrops.
- Make your characters have a conversation.
- Move your characters.
- Change something when you click on it.



Support Tinkering

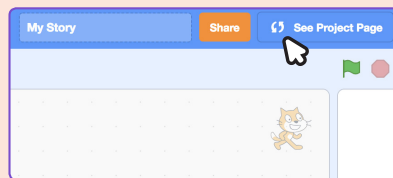
Scratch is designed to support creating by experimenting and tinkering. So, your students may want to start their stories without planning beforehand. As they create, one idea can spark another. Celebrate their sparks of creativity and the unexpected turns their stories may take.



Prepare to Share

To add instructions and credits to a project, click the button: “See project page”.

Then click the Share button if you want the project visible to others online.



Share

Help the students add their projects to a shared studio in Scratch. Give them a link to the studio. Then they can click ‘Add Projects’ at the bottom of the page.

Ask for volunteers to show their project to the group.

What's Next?

Students can use these ideas and concepts to create a variety of projects. Here are some variations on the story project you could suggest:



Retell a story

Start with a story you know and make it in Scratch. Imagine a new ending or a different setting.



Neighbourhood story

Take photos of your classroom, school, or neighborhood and use them as backdrops in your story.



Round-robin story

Give everyone 5 minutes to start a story. Then, have them switch to the next computer to add to the story. Repeat.

Created by the Scratch Team