

2022

# Elementary Integration Guide

FOURTH 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"><li>• <a href="#">Code.org</a> CS Fundamentals<ul style="list-style-type: none"><li>◦ <a href="#">4th Grade: Course E</a></li></ul></li><li>• <a href="#">Common Sense Digital Media</a></li><li>• <a href="#">Scratch</a></li><li>• <a href="#">CS First Guide</a></li><li>• <a href="#">Kahoot</a></li><li>• <a href="#">CS Unplugged</a></li></ul>
CS4MS website materials	<ul style="list-style-type: none"><li>• <a href="#">2018 Mississippi Computer Science Standards</a></li><li>• <a href="#">CS4MS Website</a></li></ul>
Keyboard practice	<ul style="list-style-type: none"><li>• <a href="#">Astro Bubbles Keyboard Practice</a></li><li>• <a href="#">Dance Mat Typing Site</a></li><li>• <a href="#">Nitrotype</a></li><li>• <a href="#">Typing.com</a></li></ul>
Teacher/student accounts	<ul style="list-style-type: none"><li>• <a href="#">Code.org</a></li><li>• <a href="#">Common Sense Digital Media</a></li><li>• <a href="#">Scratch</a></li><li>• <a href="#">CS First</a></li><li>• <a href="#">Kahoot</a></li><li>• <a href="#">CS Unplugged</a></li></ul>
For help with this guide	<ul style="list-style-type: none"><li>• Contact Mississippi State University's Center for Cyber Education: <a href="http://www.tinyurl.com/ccehelpdesk">www.tinyurl.com/ccehelpdesk</a></li></ul>



# Contents by Integrated Subjects

## English

- Week 2: RI.4.2, RL.4.2—Key Ideas and Details (Reading Literature and Informational Text)
- Week 3: RI.4.3—Key Ideas and Details (Reading Informational Text)
- Week 7: W.4.3a—Text Types and Purposes
- Week 8: RI.4.5—Craft and Structure (Reading Informational Text)
- Week 17: RL.4.1, RI.4.1—Key Ideas and Details (Reading Literature and Informational Text)
- Week 19: W.4.7—Research to Build and Present Knowledge
- Week 22: RL.4.2, RI.4.2—Key Ideas and Details (Reading Literature and Informational Text)
- Week 25: W.4.1—Text Types and Purposes
- Week 27: RL.4.4—Craft and Structure (Reading Literature)
- Week 29: W.4.3e—Text Types and Purposes
- Week 33: RI.4.9—Integration of Knowledge and Ideas (Reading Informational Text)
- Week 34: RL.4.1, RL.4.2, RL.4.3, RL.4.4—Key Ideas and Details; Craft and Structure (Reading Literature)
- Week 38: RI.4.5—Craft and Structure (Reading Informational Text)
- Week 39: RI.4.5—Craft and Structure (Reading Informational Text)
- Week 41: W.4.6—Production and Distribution of Writing

## Math

- Week 10: 4.G.1—Identify Lines and Angles and Classify Shapes
- Week 11: 4.MD.5—Understand Concepts of Angles and Measure Angles
- Week 12: 4.MD.5—Understand Concepts of Angles and Measure Angles
- Week 18: 4.OA.5—Generate and Analyze Patterns
- Week 21: 4.NBT.6—Understanding Place Value and Properties of Operations
- Week 23: 4.G.3—Identify Lines and Angles and Classify Shapes
- Week 26: 4.NBT.1—Understanding Place Value for Multi-Digit Whole Numbers
- Week 32: 4.G.1—Identify Lines and Angles and Classify Shapes
- Week 37: 4.MD.7—Understand Concepts of Angles and Measure Angles
- Week 40: 4.OA.5—Generate and Analyze Patterns

## Science

- Week 13: E.4.9.B—Earth's Systems and Cycles (Weather and Climate Patterns)
- Week 15: P.4.6A, P.4.6B, P.4.6.C—Motions, Forces, and Energy (Heat and Electricity)
- Week 19: P.4.6C.3—Motions, Forces, and Energy (Sound)
- Week 24: L.4.2—Reproduction and Heredity (Life Cycles)
- Week 30: E.4.9C.5—Earth's Systems and Cycles (Landforms and Oceans)
- Week 35: E.4.10.1—Earth's Systems and Cycles (Sources of Energy Used for Human Needs)

## Social Studies

- Week 5: Topic-G.4.3.2—Geography (Maps, Graphs, and Other Representations of Mississippi)
- Week 6: Topic-CI.4.3—Civics (Rights and Responsibilities as a Citizen of Your Community and State)
- Week 9: Topic-CI.4.3—Civics (Rights and Responsibilities as a Citizen of Your Community and State)
- Week 14: Topic-CI.4.3—Civics (Rights and Responsibilities as a Citizen of Your Community and State)
- Week 16: Topic-H.4.1—History (Symbols, Customs, and Celebrations)
- Week 20: Topic-CR.4.1.1-2—Civil Rights (Social, Political, and Economic Impact)
- Week 25: Topic-H.4.4.1—History (Literature, The Arts, Architecture, and Music in Mississippi)
- Week 28: Topic-CI.4.3.1—Civics (Rights and Responsibilities as a Citizen of Your Community and State)
- Week 31: Topic-CR.4.1.1—Civil Rights (Social, Political, and Economic Impact)
- Week 36: Topic-G.4.1.1, G.4.2.3—Geography (Physical Geography and Environmental Factors)

# Contents by Topics

## Coding

- Week 3
- Week 4
- Week 7
- Week 8
- Week 10
- Week 11
- Week 12
- Week 13
- 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 29
- Week 30
- Week 31
- Week 35
- Week 36
- Week 38
- Week 40

## Digital Citizenship

- Week 6
- Week 9
- Week 14
- Week 28
- Week 33

## Digital Literacy

- Week 1

## Keyboarding

- Week 1
- Use as a filler to build typing fluency
  - Resources: NitroType, Typing Club (a free product of EdClub)

## Robotics

- Week 32
- Week 34
- Week 37
- Week 39

## Unplugged

- Week 2
- Week 5
- Week 41
- Week 13

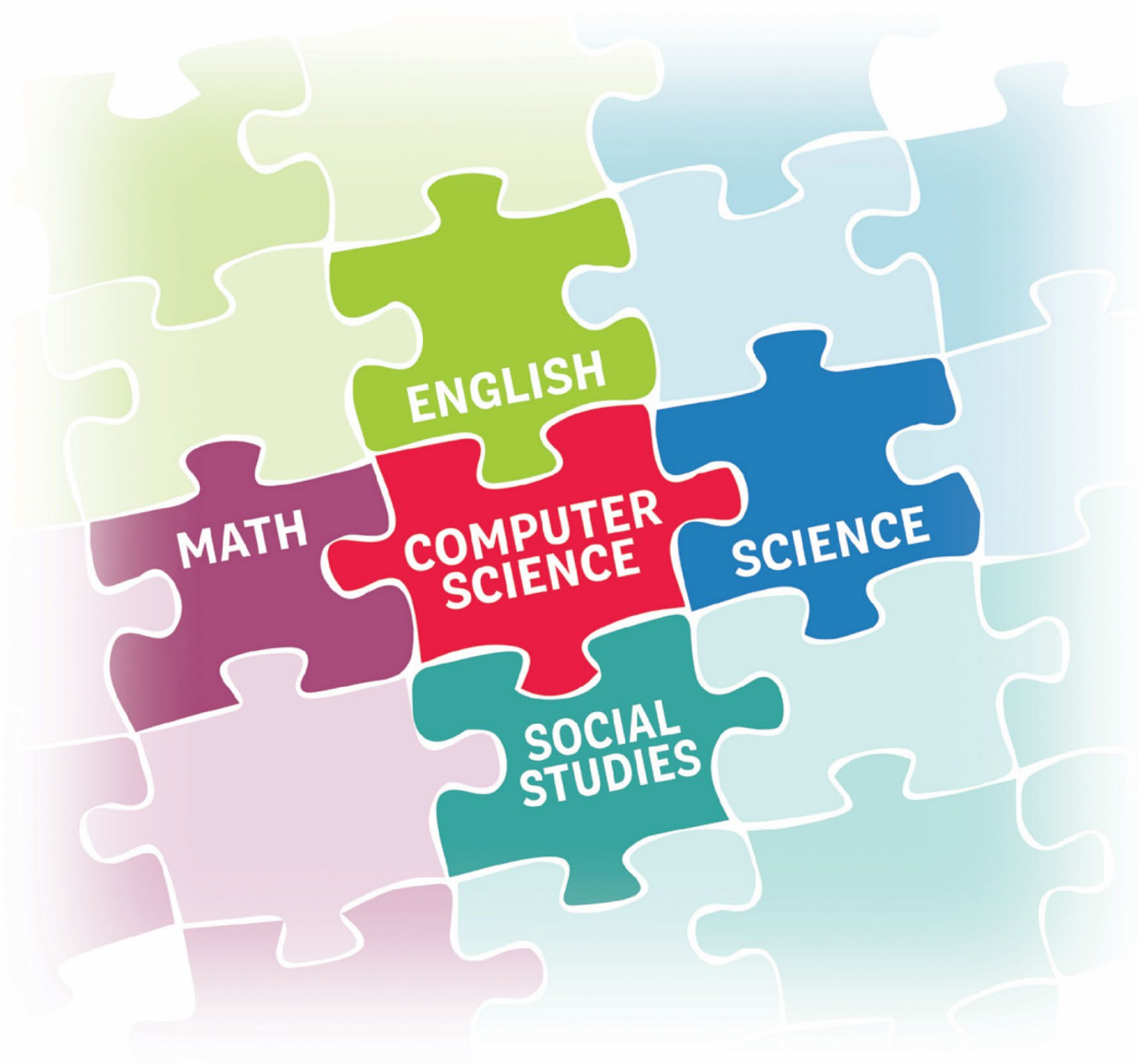
# Calendar/Pacing Per Week:

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

Week	Title	Topics	CS Standard	Standard	Subject Integrated
1	Keyboarding Basics	Digital Citizenship	CS.1B.1	None	None
2	Code.org: Graph Paper Programming → Account creation needed	Unplugged	AP.1B.4	RI.4.2 RL.4.2	ELA
3	Code.org: Introduction to Online Puzzles → Account creation needed	Unplugged	AP.1B.4	RI.4.3	ELA
4	Code.org: Follow the Algorithm	Unplugged	AP.1B.5	L.4.3a	ELA
5	Programming Using Maps	Unplugged	AP.1B.1a	G.4.3.2	Social Studies
6	Code.org: Be A Super Digital Citizen	Digital Citizenship	NB.1B.2	CI.4.3	Social Studies: MS Studies
7	Code.org: Swimming Fish With Sprite Lab	Coding	AP.1B.5	W.4.3a	Writing
8	Code.org: Alien Dance Party With Sprite Lab	Coding	AP.1B.5	RI.4.5	ELA
9	Code.org: Private and Personal Information	Digital Citizenship	NI.1B.2 NI.1B.2a	CI.4.3	Social Studies
10	Code.org: Drawing With Loops	Coding	AP.1B.4	4.G.1	Math
11	Code.org: Nested Loops in Maze	Coding	AP.1B.4	MD.5	Math
12	Code.org: Fancy Shapes Using Nested Loops	Coding	AP.1B.4 AP.1B.6	4.MD.5	Math
13	Code.org: Mini-Project—Design a Snowflake	Digital Citizenship	IC.1B.4 NI.1B.2	E.4.9.B	Science
14	Code.org: Mini-Project—About Me	Unplugged	AP.1B.1 AP.1B.4	CI.4.3	Social Studies
15	Interactive Presentation → Account creation needed	Coding	AP.1B.4 AP.1B.6	P.4.6a P.4.6b P.4.6.c	Science
16	Code.org: Songwriting	Coding	AP.1B.1 AP.1B.1a AP.1B.4 AP.1B.4a	H.4.1	Social Studies
17	Code.org: Sequential Order Using Scratch	Coding	AP.1B.10a AP.1B.10	RL.4.1 RI.4.1	ELA
18	Code.org: Functions in Minecraft	Coding	AP.1B.1 AP.1B.4	4.OA.5	Math
19	Research a Scientist Project	Digital	AP.1B.1	P.4.6c.3	Science,

		Literacy	AP.1B.4	W.4.7	Writing
20	Civil Rights History	Digital Literacy	AP.1B.3a	CR.4.1.1-2	Social Studies
21	Animating the Steps of Long-Division	Coding	AP.1B.1 AP.1B.4	4.NBT.6	Math
22	Theme, Main Idea, and Details	Coding	AP.1B.10a 1B.AP.10	RL.4.2 RI.4.2	ELA
23	Code.org: Functions With Artists	Coding	AP.1B.1 AP.1B.4	4.MD.5	Math
24	Life Cycles	Digital Literacy	AP.1B.10a AP.1B.10	L.4.2	Science
25	CSFirst: Code Your Hero	Coding	AP.1B.3a AP.1B.6b	4.W.1	Writing
26	Place Value Remix	Digital Literacy	AP.1B.1 AP.1B.4	4.NBT.1 4.NBT.4	Math
27	Literal and Figurative Language	Coding	AP.1B.10 AP.1B.11	RL.4.4	ELA
28	Digital Sharing	Digital Citizenship	AP.1B.5 AP.1B.5a	CI.4.3.1	Social Studies
29	Code.org: Conditionals in Minecraft Voyage Aquatic	Coding	AP.1B.3	4.NBT.5	Math
30	Weather Safety	Coding	AP.1B.10 AP.1B.11	E.4.9C.5	Science
31	Famous Mississippians	Coding	AP.1B.3a AP.1B.6b	CR.4.1.1	Social Studies
32	Points, Lines, Segments, Rays, and Angles	Robotics	AP.1B.3	4.G.1	Math
33	Digital Sharing Animation	Digital Citizenship	IC.1B.21 NI.1B.5	RI.4.9	ELA
34	Maze Mat Race	Robotics	AP.1B.1 AP.1B.4	RL.4.1 RL.4.2 RL.4.3 RL.4.4	ELA
35	Energy Sources	Coding	AP.1B.5 AP.1B.6	E.4.10.1 E.4.10.2	Science
36	Code.org, Conditionals With the Farmer	Coding	AP.1B.3 AP.1B.3a AP.1B.4 AP.1B.4a	G.4.1.1 G.4.2.3	Social Studies
37	Angle Measures	Robotics	AP.1B.4	4.MD.7	Math
38	Text Structures: Coding	Coding	AP.1B.4	RI.4.5	ELA
39	Text Structures: Robotics	Robotics	AP.1B.1 AP.1B.4	RI.4.5	ELA

40	Code.org, Functions With Harvester	Coding	AP.1B.5 AP.1B.6 CS.1B.3 IC.1B.1 IC.1B.2 IC.1B.3	4.OA.5	Math
41	Code.org, Designing for Accessibility *End-of-Course Project*	Digital Citizenship, Coding, Digital Literacy	AP.1B.5 AP.1B.6 CS.1B.3 IC.1B.1 IC.1B.2 IC.1B.3	W.4.6	Writing




# **Lessons and Activities**

---


FOURTH GRADE

## Week 1: Keyboarding Basics Lesson

<p>Lesson overview:</p> 	<p><b>Purpose:</b> In this keyboarding lesson, students will practice exhibiting the proper techniques for keyboarding using various instructional engagement media, such as videos and an online keyboarding game with various levels of keyboarding-skill mastery.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction             <ul style="list-style-type: none"> <li>○ <a href="#">Proper Sitting Postures on Computer</a></li> </ul> </li> <li>● Keyboarding Practice             <ul style="list-style-type: none"> <li>○ <a href="#">Keyboard Finger Guide</a></li> <li>○ <a href="#">Dance Mat Typing Site</a></li> </ul> </li> <li>● Digital Citizenship             <ul style="list-style-type: none"> <li>○ <a href="#">Keeping Games Fun and Friendly</a></li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<p>Lesson plan</p> <ul style="list-style-type: none"> <li>● <a href="#">Intro to Keyboarding – Teacher-Created Lesson Plan Common Sense Education.pdf</a></li> <li>● <a href="#">Proper Sitting Postures on Computer</a></li> <li>● <a href="#">Dance Mat Typing Site</a></li> <li>● <a href="#">Keyboard Finger Guide</a></li> </ul> <p>Digital citizenship lesson:</p> <ul style="list-style-type: none"> <li>● <a href="#">Keeping Games Fun and Friendly</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Identify home row keys</li> <li>● Demonstrate proper sitting posture for typing</li> <li>● Identify ways to develop positive interactions while gaming online</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>CS.1B.1</b>—Describe how internal and external parts of computing devices function to form a system.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Introduction Video and Discussion <b>5 min</b></li> <li>● Main Activity <b>30 min</b></li> <li>● Digital Citizenship Activity <b>15 min</b></li> <li>● Keyboarding Practice <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
<p>Subject integrated:</p>	<p>None</p>
<p>Other standards addressed:</p>	<p>None</p>
<p>Vocabulary:</p>	<p><u>Home row keys:</u> The home keys are where you place your fingers when you are learning to type. The home keys include F, D, S, and A on the left of the keyboard, and J, K, L, and ; (semicolon) on the right of the keyboard.</p>
<p>Notes:</p>	

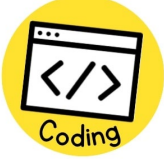


## Week 2: Code.org, Course D, Lesson 2—Graph Paper Programming

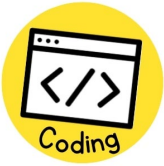
<p>Lesson overview:</p> <div style="text-align: center; margin-top: 20px;">  </div>	<p><b>Purpose:</b> The goal of this activity is to build critical thinking skills and excitement for the course, while introducing some of the fundamental programming concepts that will be used throughout the course. By introducing basic concepts like sequencing and algorithms to the class in an unplugged activity, students who are intimidated by computers can still build a foundation of understanding on these topics. In this lesson, students will learn how to develop an algorithm and encode it into a program.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction             <ul style="list-style-type: none"> <li>○ Play one of the three videos to show students the type of things robots can do.</li> </ul> </li> <li>● Graph Paper Programming             <ul style="list-style-type: none"> <li>○ In this activity, students will act as both programmers and robots, coloring in squares according to programs that they have written for one another.</li> </ul> </li> <li>● Reflection             <ul style="list-style-type: none"> <li>○ What was today's lesson about?</li> <li>○ How did you feel during today's lesson?</li> <li>○ Draw another image that you could code. Can you write the program that goes with this drawing?</li> <li>○ What other types of robots could we program if we changed what the arrows meant?</li> </ul> </li> <li>● Assessment             <ul style="list-style-type: none"> <li>○ Independent practice writing and reading a program</li> </ul> </li> <li>● Keyboarding             <ul style="list-style-type: none"> <li>○ If time remains, have students use resources from above to practice keyboarding.</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">Course D Lesson 2: Graph Paper Programming</a></li> <li>● <a href="#">Common Sense: Our Responsibilities Online</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Explain constraints of translating problems from human language to machine language</li> <li>● Reframe a sequence of steps as an encoded program</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.4</b>—Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up/Introduction <b>10 min</b></li> <li>● Main Activity <b>20 min</b></li> <li>● Wrap Up <b>15 min</b></li> <li>● Digital Citizenship Topic <b>15 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> <li>● <a href="#">Graph Paper Programming</a> - Lesson in action video</li> <li>● <a href="#">Graph Paper Programming</a> - Worksheet answer key</li> <li>● <a href="#">Graph Paper Programming</a> - Assessment answer key</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> <li>● <a href="#">Graph Paper Programming</a> - Activity worksheet</li> </ul>

	<ul style="list-style-type: none"> <li>• <a href="#">Graph Paper Programming</a> - Unplugged video (<a href="#">Download</a>)</li> <li>• <a href="#">Graph Paper Programming</a> – Assessment</li> </ul>
Subject integrated:	ELA
Other standards addressed:	<ul style="list-style-type: none"> <li>• <b>RI 4.2</b>—Determine the main idea of a text and explain how it is supported by key details; summarize the text.</li> <li>• <b>RL4.2</b>—Determine a theme of a story, drama, or poem from details in the text; summarize the text.</li> </ul>
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>
Notes:	<p>→ Teachers will need to create FREE teacher and/or student accounts  <a href="https://studio.code.org/users/sign_in">https://studio.code.org/users/sign_in</a></p>

**Week 3: Code.org, Course D, Lesson 3—Introduction to Online Puzzles**


<p>Lesson overview:</p> 	<p><b>Purpose:</b> Every classroom has a spectrum of understanding for every subject. Some students in your class may be computer wizards, while others have not had much experience at all. In order to create an equal playing (and learning) field, Code.org has developed this "Ramp Up Stage" for Course D. This can be used as either an introduction or a review of how to use Code.org and basic computer science concepts. This stage covers all prerequisites needed to start Course D.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Vocabulary words</li> </ul> </li> <li>● Preview of online puzzles as a class <ul style="list-style-type: none"> <li>○ Bring the unplugged activity from the previous lesson to an online platform.</li> </ul> </li> <li>● Online Puzzles <ul style="list-style-type: none"> <li>○ Introduction to programming with blocks</li> </ul> </li> <li>● Reflection</li> <li>● Digital Citizenship (Optional) <ul style="list-style-type: none"> <li>○ The words we choose</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">Course D Lesson 3: Introduction to Online Puzzles</a></li> <li>● <a href="#">Common Sense: The Words We Choose</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Break down a long sequence of instructions into the largest repeatable sequence</li> <li>● Modify an existing program to solve errors</li> <li>● Order movement commands as sequential steps in a program</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.4</b>—Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up/Introduction <b>10 min</b></li> <li>● Bridging Activities <b>10 min</b></li> <li>● Main Activity <b>30 min</b></li> <li>● Wrap Up <b>10 min</b></li> <li>● Digital Citizenship Topic (optional) <b>15 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> <li>● <a href="#">Pair Programming</a> - student video</li> </ul>
<p>Subject integrated:</p>	<p>ELA</p>
<p>Other standards addressed:</p>	<p><b>RI.4.3</b>—Explain events, procedures, ideas, or concepts in a historical, scientific, or technical text, including what happened and why, based on specific information in the text.</p>
<p>Vocabulary:</p>	<p><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>Loop</u>: The action of doing something over and over again  <u>Program</u>: An algorithm that has been coded into something that can be</p>

	<p>run by a machine <u>Programming</u>: The art of creating a program <u>Technical text</u>: Instructions telling us how to use or do something</p>
Notes:	<p>→Teachers will need to create free teacher and/or student accounts (when applicable) at <a href="https://www.commonsense.org/education/">https://www.commonsense.org/education/</a></p>

<p>Lesson overview:</p> 	<p><b>Purpose:</b> This lesson is designed to prepare students to think about one of the core programming concepts in Sprite Lab: Behaviors.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>• Introduction <ul style="list-style-type: none"> <li>◦ Think-pair-share</li> </ul> </li> <li>• Follow the Algorithm <ul style="list-style-type: none"> <li>◦ Unplugged activity</li> <li>◦ Discussion</li> </ul> </li> <li>• Reflection <ul style="list-style-type: none"> <li>◦ What did we learn?</li> </ul> </li> <li>• Keyboarding Practice <ul style="list-style-type: none"> <li>◦ If time remains, have students use resources from above to practice keyboarding.</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>• <a href="#">Course E Lesson 1: Follow the Algorithm</a></li> <li>• <a href="#">Common Sense: My Media Balance</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>• Begin making connections about a new way to write programs</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>• <b>AP.1B.5</b> Modify, remix, or incorporate portions of an existing program into one's own work to develop something new or add more advanced features.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>• Warm Up/Introduction <b>10 min</b></li> <li>• Main Activity <b>20 min</b></li> <li>• Wrap Up <b>15 min</b></li> <li>• Keyboarding Practice <b>15 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>• Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>• Student devices with access to the internet</li> <li>• <a href="#">Follow the Algorithm</a> - Slides</li> </ul>
<p>Subject integrated:</p>	<p>ELA</p>
<p>Other standards addressed:</p>	<p><b>L.4.3a</b>—Choose words and phrases to convey ideas precisely.</p>
<p>Vocabulary:</p>	<p><u>Behavior</u>: An action that a sprite performs continuously until it is told to stop  <u>Algorithm</u>: A list of steps to finish a task</p>
<p>Notes:</p>	


<b><u>Week 5: Programming Using Maps</u></b>	
<p>Lesson overview:</p>	<p><b><u>Purpose:</u></b></p>



	<p>In this context-setting lesson, students will play a game intended to get them thinking about sequential instructions.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● The teacher will provide the students with a map of Mississippi on chart paper or a large poster board. The students will locate their city and then give directions to points of interest throughout the state (Jackson, Ross Barnett Reservoir, MS Gulf Coast, Natchez, Vicksburg, etc.) or to the hometowns of famous Mississippians.</li> <li>● The students will use the square sticky notes as a “grid” to mark how many moves should be made in order to reach their location from their starting point.</li> <li>● The students may use the coding cards (see resources) to map out their path.</li> <li>● Keyboarding Practice             <ul style="list-style-type: none"> <li>○ If time remains, have students use resources from above to practice keyboarding.</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Coding Cards (arrows)</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Begin to make connections about a new way to write programs</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1B1a</b> students should be able to look at different ways to solve the same task and decide which would be the best solution.</li> <li>● <b>AP.1B.5</b> Modify, remix, or incorporate portions of an existing program into one’s own work to develop something new or add more advanced features.</li> <li>● <b>AP.1B.5a</b> Students should be able to modify and/or reuse portions of an existing program into their own work to create something new.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up/Introduction <b>10 min</b></li> <li>● Main Activity <b>20 min</b></li> <li>● Wrap Up <b>15 min</b></li> <li>● Keyboarding Practice <b>15 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Chart paper</li> <li>● Markers</li> <li>● Map of Mississippi</li> <li>● Small square sticky notes</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Chart paper</li> <li>● Markers</li> <li>● Map of Mississippi</li> <li>● Small square sticky notes</li> </ul>
<p>Subject integrated:</p>	<p>Social Studies</p>
<p>Other standards addressed</p>	<p><b>G.4.3.2</b>—Recognize maps, graphs, and other representations of Mississippi.</p>

Vocabulary:	<u>Behavior</u> : An action that a sprite performs continuously until it is told to stop <u>Algorithm</u> : A list of steps to finish a task
Notes:	

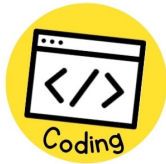


<p>Lesson overview:</p> 	<p><b>Purpose:</b> Common Sense's Digital Citizenship Curriculum addresses six core topics, based on the latest research on children, media, and technology. This lesson focuses on Cyberbullying, Digital Drama &amp; Hate Speech. Students take on these tough topics and play the active role of upstanders to build positive, supportive online communities and combat online cruelty.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Secret Superhero <ul style="list-style-type: none"> <li>○ Key vocabulary</li> </ul> </li> <li>● Being an Upstander <ul style="list-style-type: none"> <li>○ Discuss the meaning of being a cyberbully</li> <li>○ <a href="#">What Would a Super Digital Citizen Do? Student Handout</a></li> </ul> </li> <li>● Create Your Digital Superhero <ul style="list-style-type: none"> <li>○ <a href="#">Digital Citizen Superhero Student Handout</a></li> <li>○ <a href="#">Marvel's Create Your Own Superhero</a></li> </ul> </li> <li>● Save the Day Comic Strip <ul style="list-style-type: none"> <li>○ <a href="#">Classroom-Friendly Websites and Apps for Making Comics</a></li> </ul> </li> <li>● Keyboarding Practice <ul style="list-style-type: none"> <li>○ If time remains, have students use resources from above to practice keyboarding.</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Course E Lesson 4: Be A Super Digital Citizen</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Define and recognize various forms of cyberbullying</li> <li>● Explain ways that cyberbullying can be addressed</li> </ul> <p>Standards</p> <ul style="list-style-type: none"> <li>● <b>NB.1B.2</b> Discuss real-world cybersecurity problems and how personal information can be protected (cybersecurity).</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up/Introduction <b>5 min</b></li> <li>● Main Activity: Learn <b>10 min</b></li> <li>● Main Activity: Create <b>15 min</b></li> <li>● Wrap Up <b>15 min</b></li> <li>● Keyboarding (See "other" resources) <b>15 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> <li>● <a href="#">Be A Super Digital Citizen: Lesson Slides</a></li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> <li>● <a href="#">Be A Super Digital Citizen: Super Digital Citizen</a> (<a href="#">Download</a>)</li> </ul>
<p>Subject integrated:</p>	<p>Social Studies</p>
<p>Other standards addressed:</p>	<p><b>CI:4.3</b>—Identify rights and responsibilities as a citizen of your community or state.</p>
<p>Vocabulary:</p>	<p><u>Cyberbullying</u>: Using digital devices, sites, and apps to intimidate, harm, and upset someone</p> <p><u>Digital citizen</u>: Someone who uses technology responsibly to learn,</p>

	create, and participate <u>Upstander</u> : A person who supports and stands up for someone else
Notes:	

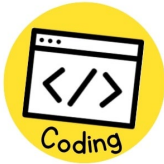
**Week 7: Code.org, Course E, Lesson 2—Swimming Fish With Sprite Lab**

Lesson overview:	<u>Purpose</u> :
------------------	------------------




	<p>This lesson is designed to introduce students to the core vocabulary of Sprite Lab and allow them to apply concepts they learned in other environments to this tool. By creating a fish tank, students will begin to form an understanding of the programming model of this tool and explore ways they can use it to express themselves.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"><li>● Introduction<ul style="list-style-type: none"><li>○ What is a sprite?</li><li>○ A sprite is controlled by a program</li></ul></li><li>● Swimming Fish Teacher Sandbox<ul style="list-style-type: none"><li>○ Model writing programs for the sprite</li></ul></li><li>● Swimming Fish with Sprite Lab<ul style="list-style-type: none"><li>○ Students will move through the lessons to create their own fish tank</li></ul></li><li>● Reflection</li><li>● Keyboarding Practice<ul style="list-style-type: none"><li>○ If time remains, have students use resources from above to practice keyboarding.</li></ul></li></ul>
Lesson links/resources:	<a href="#">Course E Lesson 2: Swimming Fish With Sprite Lab</a>
CS standards addressed:	<p>Students will be able to:</p> <ul style="list-style-type: none"><li>● Create new sprites and assign them costumes and behaviors</li><li>● Define "sprite" as a character or object on the screen that can be moved and changed</li></ul> <p>Standards:</p> <ul style="list-style-type: none"><li>● <b>AP.1B.5</b>—Modify, remix, or incorporate portions of an existing program into one's own work to develop something new or add more advanced features.</li></ul>
Time needed:	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"><li>● Warm Up/Introduction <b>10 min</b></li><li>● Bridging Activity <b>10 min</b></li><li>● Main Activity <b>20 min</b></li><li>● Wrap Up <b>15 min</b></li><li>● Keyboarding Practice <b>5 min</b></li></ul>
Materials needed:	<p>Teachers:</p> <ul style="list-style-type: none"><li>● Smartboard/projector with sound</li><li>● <a href="#">Swimming Fish Teacher Sandbox</a> - Programming level</li></ul> <p>Students:</p> <ul style="list-style-type: none"><li>● Student devices with access to the internet</li><li>● <a href="#">Sprite Lab Documentation</a> – Resource</li></ul>
Subject integrated:	Writing
Other standards addressed:	<p><b>W.4.3</b>—Write narratives to develop real or imagined experiences or events using effective technique, descriptive details, and clear event sequences.</p>
Vocabulary:	<p><u>Behavior</u>: An action that a sprite performs continuously until it is told to stop</p> <p><u>Sprite</u>: A graphic on the screen with a location, size, and appearance</p>

Notes:

<p>Lesson overview:</p> 	<p><b>Purpose:</b> This lesson offers a great introduction to events in programming and even gives students a chance to show creativity! At the end of the lesson, students will be presented with the opportunity to share their projects.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ A series of events</li> <li>○ New vocabulary</li> </ul> </li> <li>● Alien Dance Party with Sprite Lab <ul style="list-style-type: none"> <li>○ Prediction</li> <li>○ Sprites in action</li> <li>○ Skill building</li> <li>○ Free play</li> </ul> </li> <li>● Reflection</li> <li>● Free Play <ul style="list-style-type: none"> <li>○ If time remains, students can free play in their Sprite Lab to experiment different programs</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Course E Lesson 3: Alien Dance Party With Sprite Lab</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Create an interactive animation using sprites, behaviors, and events</li> <li>● Identify actions that correlate to input events</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.5</b>—Modify, remix, or incorporate portions of an existing program into one's own work to develop something new or add more advanced features.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up/Introduction <b>5 min</b></li> <li>● Main Activity <b>30 min</b></li> <li>● Wrap Up <b>10 min</b></li> <li>● Free Play <b>15 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> <li>● <a href="#">Sprite Lab Documentation</a> – Resource</li> </ul>
<p>Subject integrated:</p>	<p>ELA</p>
<p>Other standards addressed:</p>	<p><b>RI 4.5</b>—Describe the overall structure (e.g., chronology, comparison, cause/effect, problem/solution) of events, ideas, concepts, or information in a text or part of a text.</p>
<p>Vocabulary:</p>	<p><u>Event</u>: An action that causes something to happen</p>
<p>Notes:</p>	

**Week 9: Code.org, Course E, Lesson 5—Private and Personal Information**

<p>Lesson overview:</p> 	<p><b>Purpose:</b> Common Sense's Digital Citizenship Curriculum addresses six core topics based on the latest research on children, media, and technology. This lesson focuses on privacy and security. Students learn how to protect personal information and gain a deeper understanding of their data privacy rights so they can advocate for themselves and others.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Stand Up, Sit Down <ul style="list-style-type: none"> <li>○ Key Vocabulary</li> </ul> </li> <li>● Why Do People Share? <ul style="list-style-type: none"> <li>○ Learn how much is safe to share online</li> </ul> </li> <li>● Private or Personal? <ul style="list-style-type: none"> <li>○ Learn what is safe to share online</li> </ul> </li> <li>● Assessment <ul style="list-style-type: none"> <li>○ <a href="#">Private and Personal Information: Lesson Quiz</a></li> </ul> </li> <li>● Exit Ticket <ul style="list-style-type: none"> <li>○ <a href="#">Private and Personal Information: Exit Ticket</a></li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Course E Lesson 5: Private and Personal Information</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Explain the difference between private and personal information</li> <li>● Explain why it is risky to share private information online</li> <li>● Identify the reasons why people share information about themselves online</li> </ul> <p>Standard:</p> <ul style="list-style-type: none"> <li>● <b>NI.1B.2</b>—Discuss real-world cybersecurity problems and how personal information can be protected.</li> <li>● <b>NI.1B.2a</b>—Students should be able to explain what passwords are, why we use them, and use strong passwords to protect devices and information from unauthorized access.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up/Introduction <b>10 min</b></li> <li>● Analyze: Why people share? <b>10 min</b></li> <li>● Analyze: Private or personal? <b>15 min</b></li> <li>● Assessment <b>15 min</b></li> <li>● Wrap Up/Exit Ticket <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>For the teachers</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> <li>● <a href="#">Private and Personal Information: Lesson Quiz</a>-Answer key</li> <li>● <a href="#">Private and Personal Information: Lesson Slides</a>-Slide deck</li> </ul> <p>For the students</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> <li>● <a href="#">Private and Personal Information: Exit Ticket</a>-Student handout</li> <li>● <a href="#">Private and Personal Information: Lesson Quiz</a>-Form</li> <li>● <a href="#">Private and Personal Information: Private and Personal Information</a>-Student video (<a href="#">Download</a>)</li> </ul>
<p>Subject integrated:</p>	<p>Social Studies</p>
<p>Other standards addressed:</p>	<p><b>CI:4.3</b>—Identify rights and responsibilities as a citizen of your community or state.</p>
<p>Vocabulary:</p>	<p><u>Hardwired</u>: Something you are born with</p>

	<p><u>Personal Information</u>: Information about you that cannot be used to identify you because it is also true for many other people (e.g., your hair color or the city you live in)</p> <p><u>Private Information</u>: Information about you that can be used to identify you because it is unique to you (e.g., your full name or your address)</p> <p><u>Register (Online)</u>: To enter your information to sign up and get access to a website or app</p>
Notes:	

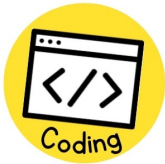
**Week 10: Code.org, Course E, Lesson 8—Drawing With Loops**

Lesson overview:

**Purpose:**

Students will practice using loops, a concept that will be revisited





throughout upcoming lessons.

**Lesson:**

- Introduction
  - Play through a puzzle to demonstrate the artist tool
- Drawing with Loops
  - Artist intro with JR Hildebrande (video)
  - Skill building
  - Loops with the artist (video)
  - Skill building
  - Challenge
  - Practice
  - Challenge
  - Extra lessons
- Reflection
- Keyboarding Practice
  - If time remains, have students use resources from above to practice keyboarding.

Lesson links/resources:

[Course E Lesson 8: Drawing with Loops](#)

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.

Standard:

- **AP.1B.4** Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.

Time needed:

**Total Time: 60 min**

- Warm Up/Introduction **10 min**
- Main Activity **30 min**
- Wrap Up/Reflection **15 min**
- Keyboarding Practice **5 min**

Materials needed:

Teachers:

- Smartboard/projector with sound

Students:

- Student devices with access to the internet

Subject integrated:

Math

Other standards addressed:

**4.G.1**—Draw points, lines, line segments, rays, angles (right, acute, obtuse), and perpendicular and parallel lines. Identify these in two-dimensional figures.

Vocabulary:

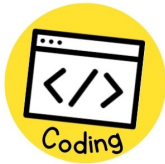
Loop: The action of doing something over and over again  
Repeat: To do something again

Notes:

**Week 11: Code.org, Course D, Lesson 9—Nested Loops in Maze**

Lesson overview:

**Purpose:**



In this introduction to nested loops, students will go outside of their comfort zone to create more efficient solutions to puzzles. In earlier puzzles, loops pushed students to recognize repetition. Here, students will learn to recognize patterns within repeated patterns to develop these nested loops. This stage starts off by encouraging students to try to solve a puzzle where the code is irritating and too complex to write out the long way. After a video introduces nested loops, students are shown an example and asked to predict what will happen when a loop is put inside of another loop. This progression leads to plenty of practice for students to solidify and build on their understanding of looping in programming.

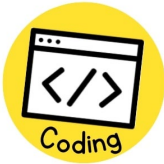
**Lesson:**

- Introduction
  - What do loops do?
  - How do we use loops?
- Nested loops in Maze
  - Skill building
  - Nested loops with the bee (video)
  - Prediction
  - Skill building
  - Challenge
  - Practice
  - Prediction
  - Lesson extras
- Reflection
- Keyboarding Practice
  - If time remains, have students use resources from above to practice keyboarding.

Lesson links/resources:	<a href="#">Course D Lesson 9: Nested Loops in Maze</a>
CS standards addressed:	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Break complex tasks into smaller, repeatable sections</li> <li>● Identify the benefits of using a loop structure instead of manual repetition</li> <li>● Recognize large, repeated patterns as made from smaller repeated patterns</li> </ul> <p>Standards</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.4</b>—Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.</li> </ul>
Time needed:	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up/Introduction <b>10 min</b></li> <li>● Main Activity <b>30 min</b></li> <li>● Wrap Up/Reflection <b>15 min</b></li> <li>● Keyboarding Practice <b>5 min</b></li> </ul>
Materials needed:	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
Subject integrated:	Math
Other standards	<b>4.MD.5</b> —Recognize angles as geometric shapes that are formed

addressed:	wherever two rays share a common endpoint and understand concepts of angle measurement.
Vocabulary:	<u>Loop</u> : The action of doing something over, and over again <u>Repeat</u> : To do something again
Notes:	

**Week 12: Code.org, Course E, Lesson 9—Fancy Shapes Using Nested Loops**

<p>Lesson overview:</p> 	<p><b>Purpose:</b> The purpose of this activity is to utilize nested loops to inspire students with artistic minds to see coding as another creative outlet. This set of puzzles was built to develop critical thinking skills, an understanding of elementary geometry, and creativity—all within the scope of nested loops!</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Review Code.org, Course D-Lesson 11 (Week #10)</li> </ul> </li> <li>● Fancy Shapes Using Nested Loops <ul style="list-style-type: none"> <li>○ Skill building</li> <li>○ Nested loops (video)</li> <li>○ Skill building</li> <li>○ Challenge</li> <li>○ Practice</li> <li>○ Free play</li> <li>○ Lesson extras</li> </ul> </li> <li>● Reflection</li> <li>● Keyboarding Practice <ul style="list-style-type: none"> <li>○ If time remains, have students use resources from above to practice keyboarding.</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Course E Lesson 9: Fancy Shapes Using Nested Loops</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Break complex tasks into smaller repeatable sections</li> <li>● Combine simple shapes into complex designs with nested loops</li> <li>● Count the number of times an action should be repeated and represent it as a loop</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.4</b>—Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up/Introduction <b>10 min</b></li> <li>● Main Activity <b>30 min</b></li> <li>● Wrap Up/Reflection <b>15 min</b></li> <li>● Keyboarding Practice <b>5 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> <li>● <a href="#">Pause and Think Online</a>-Video</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> <li>● <a href="#">Turns &amp; Angles</a>-Student handout</li> <li>● <a href="#">Turns &amp; Angles</a>-Student video</li> </ul>
<p>Subject integrated:</p>	<p>Math</p>
<p>Other standards addressed:</p>	<p><b>4. MD.5</b>—Recognize angles as geometric shapes that are formed wherever two rays share a common endpoint and understand concepts of angle measurement.</p>
<p>Vocabulary:</p>	<p><u>Loop</u>: The action of doing something over and over again <u>Repeat</u>: To do something again</p>

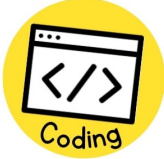
Notes:

**Week 13: Code.org, Course E, Lesson 10—Mini-Project: Design a Snowflake**

Lesson overview:

**Purpose:**

In this lesson, students will get to apply their skills with nested loops to

	<p>create images that they will be excited to share.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Discuss the last set of puzzles</li> </ul> </li> <li>● Nested Loops with Frozen <ul style="list-style-type: none"> <li>○ Mini-Project: Snowflake #1</li> <li>○ Mini-Project: Snowflake #2</li> </ul> </li> <li>● Reflection</li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Course E Lesson 10: Mini-Project: Design a Snowflake</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Break apart code into the largest repeatable sequences using both loops and nested loops</li> <li>● Describe when a loop, nested loop, or no loop is needed</li> <li>● Recognize the difference between using a loop and a nested loop</li> </ul> <p>Standards</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.4</b>—Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.</li> <li>● <b>AP.1B.6</b>—Use an iterative process to plan the development of a program by including others' perspectives and considering user preferences.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up/Introduction <b>15 min</b></li> <li>● Main Activity <b>30 min</b></li> <li>● Wrap Up/Reflection <b>15 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to internet</li> </ul>
<p>Subject integrated:</p>	<p>Science</p>
<p>Other standards addressed:</p>	<p><b>E.4.9.B</b>—Students will demonstrate an understanding of weather and climate patterns.</p>
<p>Vocabulary:</p>	<p><u>Loop</u>: The action of doing something over and over again  <u>Repeat</u>: To do something again</p>
<p>Notes:</p>	

### Week 14: Code.org, Course E, Lesson 6—Mini-Project: About Me

<p>Lesson overview:</p>	<p><b>Purpose:</b>  This lesson is meant to make the previous lesson on personal and private information personally relevant for students. With safe (personal) and unsafe (private) examples in mind, students practice safe self-expression on the web, using Sprite Lab to fashion their own sprite costumes and</p>
-------------------------	---



generate text.

**Lesson:**

- Introduction
  - Review personal and private information
- Interactive poster about me
  - Exploration: Rikki's poster
  - Mini-project: About me
- Reflection
  - Share out
- Keyboarding Practice
  - If time remains, have students use resources from above to practice keyboarding.

Lesson links/resources:

[Course E Lesson 6: Mini-Project: About Me](#)

CS standards addressed:

Students will be able to:

- Choose what information about themselves is safe to share online
- Create an interactive computer program that expresses who they are with text and custom images

Standards

- **IC.1B.4**—Use public domain or creative commons media and refrain from copying or using material created by others without permission.
- **NI.1B.2**—Discuss real-world cybersecurity problems and how personal information can be protected.

Time needed:

**Total Time: 60 min**

- Warm Up/Introduction **10 min**
- Main Activity **35 min**
- Wrap Up **15 min**
- Keyboard Practice **5 min**

Materials needed:

Teachers:

- Smartboard/projector with sound
- [Pause and Think Online](#)-Video

Students:

- Student devices with access to the internet
- [About Me Planning Guide](#)-Handout
- [Sprite Lab Documentation](#)-Resource

Subject integrated:

Social Studies

Other standards addressed:

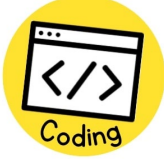
**CI.4.3**—Identify rights and responsibilities as a citizen of your community or state.

Vocabulary:

None



Notes:

<p>Lesson overview:</p> 	<p><b>Purpose:</b> Students will create an interactive presentation to explain the properties of heat, electrical, sound, and light energy.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>• Using <i>Interactive Presentation</i>, students will create a presentation (or use an existing one) and make it interactive using Scratch for CS First. Students add at least three slides, including a title slide that introduces their topic. Use this lesson to teach students how to present reports, research, or write alternative story endings in a way that engages an audience and encourages collaboration.</li> <li>• This lesson was designed for students in Grades 3 through 5 and can be adapted for many different ages and audiences. It takes approximately an hour to run.</li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">CS First: Interactive Presentation Lesson Plan</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>• Create an interactive presentation using Scratch Standards</li> <li>• <b>AP.1B.4</b>—Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.</li> <li>• <b>AP.1B.6</b>—Use an iterative process to plan the development of a program by including others' perspectives and considering user preferences.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>• Warm Up/Introduction <b>7 min</b></li> <li>• Main Activity <b>45 min</b></li> <li>• Wrap Up/Reflection <b>8 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>• Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>• Student devices with access to internet</li> </ul>
<p>Subject integrated:</p>	<p>Science</p>
<p>Other standards addressed:</p>	<ul style="list-style-type: none"> <li>• <b>P.4.6A</b>—Students will demonstrate an understanding of the common sources and uses of heat and electric energy and the materials used to transfer heat and electricity.</li> <li>• <b>P.4.6B</b>—Students will demonstrate an understanding of the properties of light as forms of energy.</li> <li>• <b>P.4.6C</b>—Students will demonstrate an understanding of the properties of sound as a form of energy.</li> </ul>
<p>Vocabulary:</p>	<p><u>Parallelism</u>: The process of events happening at the same time, either independently or interdependently</p> <p><u>Debugging</u>: The process of identifying and fixing error(s) in a program when it is not functioning as expected</p> <p><u>Control structures</u>: Sections of code that order the direction or flow of how a program functions. The control structure in this lesson is focused on loops.</p>

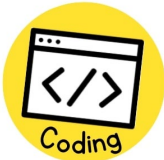
Notes:

→ Teachers will need to create FREE teacher and/or student accounts

<https://csfirst.withgoogle.com/s/en/home>

<https://scratch.mit.edu/join>

A teacher account does not have to be created to use Scratch. However, if you and your students establish a free account, your students can share their creations with you and other students.

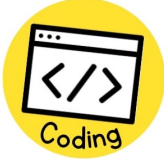
<p>Lesson overview:</p> 	<p><b>Purpose:</b> The use of functions helps simplify code and develop the student's ability to organize their program. Students will quickly recognize that writing functions can make their long programs easier to read and easier to debug if something goes wrong.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Key vocabulary</li> <li>○ Sing a song</li> </ul> </li> <li>● Songwriting <ul style="list-style-type: none"> <li>○ <a href="#">Songwriting with Functions</a> - video</li> <li>○ <a href="#">Functions Unplugged: Songwriting</a> - handout</li> </ul> </li> <li>● Assessment <ul style="list-style-type: none"> <li>○ <a href="#">Functions Unplugged: Songwriting</a> - assessment</li> </ul> </li> <li>● Reflection</li> <li>● Keyboarding Practice <ul style="list-style-type: none"> <li>○ If time remains, have students use resources from above to practice keyboarding.</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">Course E Lesson 11- Songwriting</a></li> <li>● <a href="#">The Mississippi State Song "Go, Mississippi" Lyrics</a></li> <li>● <a href="#">State Song of Mississippi- Go, Mississippi</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Describe how functions can make programs easier to write</li> <li>● Identify sections of a song to pull into a function</li> <li>● Locate repeating phrases inside song lyrics</li> </ul> <p>Standards</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.1</b>—Compare and refine multiple algorithms for the same task and determine which is the most appropriate.</li> <li>● <b>AP.1B.1a</b>—Students should be able to look at different ways to solve the same task and decide which would be the best solution.</li> <li>● <b>AP.1B.4</b>—Decompose (break down) problems into smaller, manageable subproblems to facilitate the program.</li> <li>● <b>AP.1B.4a</b>—Students should be able to break down problems into smaller, manageable tasks.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up/Introduction <b>20 min</b></li> <li>● Main Activity <b>20 min</b></li> <li>● Wrap Up <b>5 min</b></li> <li>● Keyboarding Practice <b>15 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> <li>● <a href="#">Functions Unplugged: Songwriting</a>-Lesson in action video</li> <li>● <a href="#">Functions Unplugged: Songwriting</a>-Assessment answer key for the students</li> <li>● <a href="#">Songwriting with Functions</a>-Unplugged video (<a href="#">Download</a>)</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● <a href="#">Functions Unplugged: Songwriting</a>-Worksheet</li> <li>● <a href="#">Functions Unplugged: Songwriting</a>-Assessment</li> </ul>
<p>Subject integrated:</p>	<p>Social Studies</p>
<p>Other standards addressed:</p>	<p><b>H.4.1</b>—Recognize symbols, customs, and celebrations representative of our community, Mississippi, and the United States.</p>

Vocabulary:	<u>Function</u> : A piece of code that you can easily call over and over again
Notes:	

## Week 17: Sequential Order Using Scratch

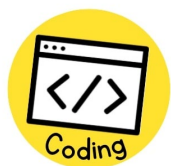
Lesson overview:

Purpose:

	<p>The students will build skills essential to reading comprehension by retelling a story through a plugged coding activity using Scratch.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>• The student will create an animation that recalls the details and events of the story chosen by the teacher, in sequential order.</li> <li>• Keyboarding Practice <ul style="list-style-type: none"> <li>◦ If time remains, have students use resources from above to practice keyboarding.</li> </ul> </li> </ul>
Lesson links/resources:	<a href="#">Scratch: Sequential Order Using Scratch</a>
CS standards addressed:	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>• Build story retelling skills through the creation of an animation.</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>• <b>AP.1B.10</b>—Students should explain code choices using comments within the code, presentations, and demonstrations.</li> <li>• <b>AP.1B.3</b>—Create programs that include sequences, events, loops, and conditionals.</li> </ul>
Time needed:	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>• Expectations/Directions <b>10 min</b></li> <li>• Creating Animation <b>40 min</b></li> <li>• Keyboarding <b>10 min</b></li> </ul>
Materials needed:	<p>Teachers:</p> <ul style="list-style-type: none"> <li>• Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>• Student devices with access to the internet</li> </ul>
Subject integrated:	ELA
Other standards addressed:	<ul style="list-style-type: none"> <li>• <b>RI.4.1</b>—Refer to details and examples in a text when explaining what the text says explicitly and when drawing inferences from the text.</li> <li>• <b>RI.4.1</b>—Refer to details and examples in a text when explaining what the text says explicitly and when drawing inferences from the text.</li> </ul>
Vocabulary:	<p><u>Behavior</u>: An action that a sprite performs continuously until it is told to stop</p> <p><u>Algorithm</u>: A list of steps to finish a task</p>
Notes:	<p>A teacher account does not have to be created to use Scratch. However, if you and your students establish a free account, your students can share their creations with you and other students.</p>

**Week 18: Code.org, Course E, Lesson 12—Functions in Minecraft**

Lesson overview:	<p><b>Purpose:</b> Students will discover the versatility of programming by practicing</p>
------------------	--



functions in different environments. Here, students will recognize reusable patterns and be able to incorporate named blocks to call predefined functions.

**Lesson:**

- Introduction
  - Key vocabulary
- Bridging Activity (choose one)
  - Unplugged activity using some blocks
  - Preview of online puzzles
- Functions in Minecraft
  - Skill building
  - Minecraft-repeat loops (video)
  - Skill building
  - Minecraft-functions
  - Skill building
  - Minecraft-congratulations
  - Free play
- Reflection

Lesson links/resources:

[Course E Lesson 12: Functions in Minecraft](#)

CS standards addressed:

Students will be able to:

- Use functions to simplify complex programs
- Use predetermined functions to complete commonly repeated tasks

Standards

- **AP.1B.1**—Compare and refine multiple algorithms for the same task and determine which is the most appropriate.
- **AP.1B.4**—Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.

Time needed:

**Total Time: 60 min**

- Warm Up **10 min**
- Bridging Activity **15 min**
- Main Activity **30 min**
- Wrap Up/Reflection **5 min**

Materials needed:

Teachers:

- Smartboard/projector with sound

Students:

- Student devices with access to the internet

Subject integrated:

Math

Other standards addressed:

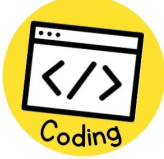
**4.OA.5**—Generate a number or shape pattern that follows a given rule. Identify apparent features of the pattern that were not explicit in the rule itself.

Vocabulary:

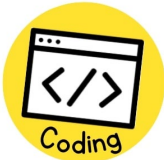
Function: A piece of code that you can easily call over and over again

Notes:

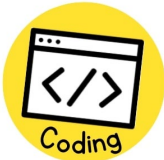


<p>Lesson overview:</p> 	<p><b>Purpose:</b> Students will research scientists and create a presentation to share with the class.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Choose a scientist</li> </ul> </li> <li>● Code Your Scientist <ul style="list-style-type: none"> <li>○ Show "Code Your Hero" video (<a href="#">CS First: Code Your Hero-Video</a>)</li> <li>○ Students create their animations in Scratch</li> </ul> </li> <li>● Reflection</li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Research a Scientist Project</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to</p> <ul style="list-style-type: none"> <li>● Use event blocks (like "when flag clicked") to trigger a series of code</li> <li>● Sequence "say" and "wait" blocks to make their hero speak and have a dialogue with another character</li> <li>● Program actions to happen using "when key pressed" events.</li> <li>● Move an object across the screen using motion blocks</li> <li>● Repeat actions using loop blocks</li> <li>● Program their hero to score points when they perform a certain action</li> <li>● Draw their own hero using the "Paint Editor" in the <i>Scratch for CS First</i> coding editor. (This add-on video pairs well with the <a href="#">Optional Planning Activity</a>.)</li> </ul> <p>Standards</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.2</b>—Create programs that use variables to store and modify data.</li> <li>● <b>AP.1B.3</b>—Create programs that include sequences, events, loops, and conditionals.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Introduction <b>3 min</b></li> <li>● Main Activity <b>50 min</b></li> <li>● Wrap Up <b>7 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
<p>Subject(s) integrated:</p>	<p>Science Writing</p>
<p>Other standards addressed:</p>	<p>Science</p> <ul style="list-style-type: none"> <li>● <b>P.4.6C.3</b>—Obtain and communicate information about scientists who pioneered in the science of sound, (e.g., Alexander Graham Bell, Robert Boyle, Daniel Bernoulli, and Guglielmo Marconi).</li> </ul> <p>Writing</p> <ul style="list-style-type: none"> <li>● <b>W.4.7</b>—Conduct short research projects that build knowledge through investigation.</li> </ul>
<p>Vocabulary:</p>	<p><u>Parallelism</u>: The process of events occurring simultaneously, either independently or interdependently</p>

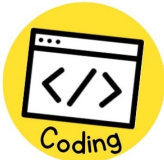
	<p><u>Debugging</u>: The process of identifying and fixing errors in a program that is not functioning as expected</p> <p><u>Control structure</u>: Sections of code that order the direction or flow of how a program functions. The control structure in this lesson is focused on loops</p> <p><u>Variable</u>: A container that stores a value that can change</p>
Notes:	

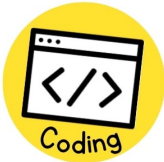
<p>Lesson overview:</p> 	<p><b>Purpose:</b> Students will talk about an important event in Civil Rights History by creating a story in Scratch. The students must choose a background and use two sprite characters in their animation.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction and expectation of assignment</li> <li>● Scratch <ul style="list-style-type: none"> <li>○ Students choose a background</li> <li>○ Students choose the sprite characters for their animation</li> <li>○ Animations should be about important Civil Rights history and/or events</li> </ul> </li> <li>● Reflection/Share</li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Scratch: Civil Rights History</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Use event blocks (like “when flag clicked”) to trigger a series of code</li> <li>● Sequence at least 3 “say” blocks between two sprites (characters) to construct a dialogue</li> <li>● Program a conditional so that the computer can decide based on a user response</li> <li>● Produce repeated movements by applying control blocks to their program</li> </ul> <p>Standard</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.3a</b>—Students should be able to create programs that include sequences, events, loops, and conditions.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Introduction <b>5 min</b></li> <li>● Civil Rights Project <b>45 min</b></li> <li>● Reflection/Share <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
<p>Subject integrated:</p>	<p>Social Studies</p>
<p>Other standards addressed:</p>	<p><b>CR.4.1.1-2</b>—Describe Mississippi’s entry into statehood.</p>
<p>Vocabulary:</p>	<p><u>Control structures</u>: Sections of code that order the direction of flow of how a program function. Control structures include conditionals and loops.</p> <p><u>Debugging</u>: The process of identifying and fixing error(s) in a program when it is not functioning as expected</p>
<p>Notes:</p>	

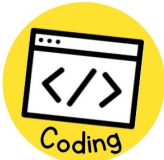
**Week 21: Animating the Steps of Long Division**

<p>Lesson overview:</p> 	<p><b>Purpose:</b> The goal of this lesson is to utilize facets of coding to facilitate accuracy and skill-building within the multi-step process of long division. Students will use block-coding to animate the mnemonic device “Does McDonald’s Sell Cheeseburgers?” which is used to help them remember the many steps of long division.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Review the steps of long division</li> <li>○ Watch the “Animate a Name” tutorial on Scratch</li> </ul> </li> <li>● Word Animation <ul style="list-style-type: none"> <li>○ Pick a background</li> <li>○ Select the desired letters</li> <li>○ Code sprite characters</li> <li>○ Run the code</li> </ul> </li> <li>● Assessment <ul style="list-style-type: none"> <li>○ Student presentations</li> </ul> </li> <li>● Wrap Up</li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Scratch: Animating the Steps of Long Division</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Look at different ways to solve the same task and decide which would be the best solution</li> <li>● Break down problems into smaller, simpler tasks</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.1</b>—Compare and refine multiple algorithms for the same task and determine which is the most appropriate. more advanced features.</li> <li>● <b>AP.1B.4</b>—Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up/Bridging <b>10 min</b></li> <li>● Main Activity <b>40 min</b></li> <li>● Wrap Up <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
<p>Subject integrated:</p>	<p>Math</p>
<p>Other standards addressed:</p>	<p><b>4.NBT.6</b>—Find whole-number quotients and remainders with up to four-digit dividends and one-digit divisors, using strategies based on place value, the properties of operations, and/or the relationship between multiplication and division. Illustrate and explain the calculation by using equations, rectangular arrays, and/or area models.</p>
<p>Vocabulary:</p>	<p><u>Algorithm</u>: A list of steps to finish a task <u>Behavior</u>: An action that a sprite performs continuously until it is told to stop</p>

	<p><u>Parallelism</u>: The process of events occurring simultaneously, either independently or interdependently</p> <p><u>Variable</u>: A container that stores a value that can change</p>
Notes:	

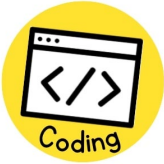
<p>Lesson overview:</p> 	<p><u>Purpose:</u> The students will build skills essential to reading comprehension. The student will determine a theme of a story, drama, or poem chosen by the teacher, or determine the main idea of a text and explain how it is supported by key details through a plugged-in coding activity (animation).</p> <p><u>Lesson:</u></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Recall theme, main idea, and details</li> <li>○ Introducing lesson activity. TTW assign a short, story, drama, or poem</li> <li>○ Provide instructions on what to include in animation</li> </ul> </li> <li>● Pre-Writing <ul style="list-style-type: none"> <li>○ Brainstorming</li> </ul> </li> <li>● Create Animation</li> <li>● Wrap Up <ul style="list-style-type: none"> <li>○ Reflection/Share</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Scratch: Theme, Main Idea, and Details</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Determine the theme or main idea of a story, drama, or poem, and explain how it is supported, through the creation of an animation</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.10a</b>—Students should explain code choices using comments within the code, presentations, and demonstrations.</li> <li>● <b>1B.AP.10</b>—Create programs that include sequences, events, loops, and conditionals.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Introduction <b>10 min</b></li> <li>● Main Activity <b>40 min</b></li> <li>● Wrap Up <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
<p>Subject integrated:</p>	<p>ELA</p>
<p>Other standards addressed:</p>	<ul style="list-style-type: none"> <li>● <b>RI.4.2</b>—Determine the theme of a story, drama, or poem from details in the text.</li> <li>● <b>RI.4.2</b>—Determine the main idea of a text and explain how it is supported by the key details.</li> </ul>
<p>Vocabulary:</p>	<p><u>Behavior:</u> An action that a sprite performs continuously until it is told to stop</p> <p><u>Algorithm:</u> A list of steps to finish a task</p>
<p>Notes:</p>	

<p>Lesson overview:</p> 	<p><b>Purpose:</b> One of the most important components to this lesson is providing students with a space to create something they are proud of. These puzzles progress to more and more complex images, but each new puzzle only builds off the previous puzzle. At the end of this lesson, students will feel confident and proud of their hard work.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Recall the use of a function</li> </ul> </li> <li>● Functions with Artist <ul style="list-style-type: none"> <li>○ Prediction</li> <li>○ Skill building</li> <li>○ Challenge</li> <li>○ Practice</li> <li>○ Lesson extras</li> </ul> </li> <li>● Reflection <ul style="list-style-type: none"> <li>○ Students will share what they felt the hardest and easiest parts of the lesson were</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Course E Lesson 13: Functions With Artist</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Categorize and generalize code into useful functions</li> <li>● Recognize when a function could help to simplify a program</li> </ul> <p>Standards</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.1</b>—Compare and refine multiple algorithms for the same task and determine which is the most appropriate.</li> <li>● <b>AP.1B.4</b>—Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up/Introduction <b>15 min</b></li> <li>● Main Activity <b>30 min</b></li> <li>● Wrap Up/Reflection <b>15 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
<p>Subject integrated:</p>	<p>Math</p>
<p>Other standards addressed:</p>	<p><b>4. MD.5</b>—Recognize angles as geometric shapes that are formed wherever two rays share a common endpoint and understand concepts of angle measurement.</p>
<p>Vocabulary:</p>	<p><u>Function</u>: A piece of code that you can easily call over and over again</p>
<p>Notes:</p>	

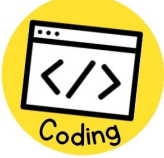
<p>Lesson overview:</p> 	<p><u>Purpose:</u> Students will create an interactive presentation to explain the life cycle of a specific plant or animal.</p> <p><u>Lesson:</u></p> <ul style="list-style-type: none"> <li>• Introduction: Review various life cycles</li> <li>• Create an animation showing the steps in chosen life cycle</li> <li>• Play the animation showing the cycle repeating in a loop</li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Scratch: Life Cycles</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>• Create an animation showing the life cycle of a chosen animal</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>• <b>AP.1B.10a</b>—Students should explain code choices using comments within the code, presentations, and demonstrations.</li> <li>• <b>1B.AP.10</b>—Create programs that include sequences, events, loops, and conditionals.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>• Warm Up/Introduction <b>10 min</b></li> <li>• Main Activity <b>35 min</b></li> <li>• Wrap Up <b>15 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>• Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>• Student devices with access to the internet</li> </ul>
<p>Subject integrated:</p>	<p>Science</p>
<p>Other standards addressed:</p>	<p><b>L.4.2</b>—Students will demonstrate an understanding of life cycles, including familiar plants and animals (e.g., reptiles, amphibians, or birds).</p>
<p>Vocabulary:</p>	<p><u>Loop:</u> The action of doing something over and over again <u>Repeat:</u> To do something again</p>
<p>Notes:</p>	



## Week 25: CS First—Code Your Hero

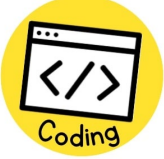
<p>Lesson overview:</p> 	<p><u>Purpose:</u> The teacher will tell the students that they are going to bring their coding to life. They will create an interactive activity by turning a famous person that they look up to into a superhero.</p> <p><u>Lesson:</u></p> <ul style="list-style-type: none"><li>● Introduction:<ul style="list-style-type: none"><li>○ Discuss the definition of hero (will likely mean different things to different people)</li><li>○ Allow students a few minutes to think of a person they look up to</li></ul></li><li>● Main Activity<ul style="list-style-type: none"><li>○ Choose your sprite</li><li>○ Choose your background/setting</li><li>○ Code your animation to discuss your famous superhero</li></ul></li><li>● Reflection/Share</li><li>● Keyboarding Practice<ul style="list-style-type: none"><li>○ If time remains, have students use resources from above to practice keyboarding.</li></ul></li></ul>
<p>Lesson links/resources:</p>	<p><a href="#">CS First: Code Your Hero</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"><li>● Create/present a program that includes sequences, loops, events, and conditionals, in the form of an animation</li></ul> <p>Standards:</p> <ul style="list-style-type: none"><li>● <b>AP.1B.3a</b> Students should be able to create programs that include sequences, events, loops, and conditionals.</li><li>● <b>AP.1B.6b</b> Students should document the plan as, for example, a storyboard, flowchart, pseudocode, or story map.</li></ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"><li>● Introduction <b>5 min</b></li><li>● Main Activity <b>40 min</b></li><li>● Wrap Up <b>5 min</b></li><li>● Keyboarding <b>10 min</b></li></ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"><li>● Smartboard/projector with sound</li></ul> <p>Students:</p> <ul style="list-style-type: none"><li>● Student devices with access to the internet</li></ul>
<p>Subject integrated:</p>	<p>ELA</p>
<p>Other standards addressed:</p>	<p><b>W.4.1</b>—Write opinion pieces on topics or texts, supporting a point of view with reasons and information.</p>
<p>Vocabulary:</p>	<p><u>Debugging</u>: The process of identifying and fixing errors in a program that is not functioning as expected</p>

	<p><u>Control structures</u>: Sections of code that order the direction of flow of how a program function. Control structures include conditionals and loops</p> <p><u>Variable</u>: A container that stores a value that can change</p>
Notes:	


<p>Lesson overview:</p> 	<p><b>Purpose:</b> The goal of this lesson is to utilize facets of coding to facilitate accuracy and skill-building within the multi-step process of subtracting across zeros. Students will use block-coding to animate their knowledge of the steps to accurately subtract across zeros to derive the difference of two numbers.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Recall steps of subtracting across zeros</li> <li>○ Discuss remixing</li> </ul> </li> <li>● Activity (Remixing) <ul style="list-style-type: none"> <li>○ Background</li> <li>○ Sprite</li> <li>○ Animations <ul style="list-style-type: none"> <li>■ Explain how to subtract across zeros within</li> </ul> </li> <li>○ Run the code</li> </ul> </li> <li>● Assessment <ul style="list-style-type: none"> <li>○ Students “teaching” with their animations</li> </ul> </li> <li>● Reflection/Share</li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Scratch: Subtracting Across Zeros</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Modify and/or reuse portions of an existing program into their own work to create something new</li> <li>● Work collaboratively to compare and refine multiple algorithms to produce a cohesive, effective presentation</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.1</b>—Compare and refine multiple algorithms for the same task and determine which is the most appropriate.</li> <li>● <b>AP.1B.5</b>—Modify, remix, or incorporate portions of an existing program into one’s own work to develop something new or add more advanced features.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Introduction <b>5 min</b></li> <li>● Activity <b>35 min</b></li> <li>● Assessment <b>10 min</b></li> <li>● Reflection/Share <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
<p>Subject integrated:</p>	<p>Math</p>
<p>Other standards addressed:</p>	<p><b>4.NBT.1</b>—Explain that in a multi-digit whole number, a digit in one place represents 10 times as much as it represents in the place to its right, up to 100,000.</p> <p><b>4.NBT.4</b>—Fluently add and subtract (including subtracting across zeros) multi-digit whole numbers using the standard algorithm.</p>
<p>Vocabulary:</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:

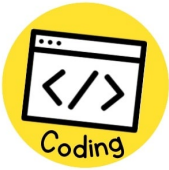
**Week 27: Literal and Figurative Language**

<p>Lesson overview:</p> 	<p><u>Purpose:</u> This lesson plan is designed to help you teach figurative language in a fun, visual, and engaging way through coding.</p> <p><u>Lesson:</u></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Review figurative language: Metaphors, similes, personification, hyperbole, and idioms are just some types of figurative language.</li> </ul> </li> <li>● Show the <i>Figurative Language-Example Project (Simile)</i></li> <li>● Pre-writing (Brainstorming)</li> <li>● Create Animation</li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">CS First: Literal and Figurative Language</a></li> <li>● <a href="#">Scratch: Figurative Language Examples</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Create programs that include sequences, events, loops, and conditionals</li> <li>● Decompose problems into smaller problems</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.10</b> Create programs that include sequences, events, loops, and conditionals.</li> <li>● <b>AP.1B.11</b> Decompose problems into smaller, manageable subproblems to facilitate the program development process.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Introduction <b>10 min</b></li> <li>● Main Activity <b>40 min</b></li> <li>● Wrap Up <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
<p>Subject integrated:</p>	<p>ELA</p>
<p>Other standards addressed:</p>	<p><b>RL.4.4</b>—Determine the meaning of words and phrases as they are used in a text, including those that allude to significant characters found in mythology (e.g., Herculean).</p>
<p>Vocabulary:</p>	<p><u>Parallelism</u>: The process of events happening at the same time, either independently or interdependently</p> <p><u>Debugging</u>: The process of identifying and fixing error(s) in a program when it is not functioning as expected</p> <p><u>Control structures</u>: Sections of code that order the direction or flow of how a program functions. The control structure in this lesson focuses on loops.</p>
<p>Notes:</p>	

**Week 28: Code.org, Course E, Lesson 7: Digital Sharing**

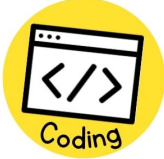
<p>Lesson overview:</p> 	<p><b>Purpose:</b> This exploratory lesson helps students understand the challenges and benefits of respecting ownership and copyright, particularly in digital environments. Students will soon be creating projects to share and most of these projects will contain either code or imagery that students did not create themselves. This lesson is here to show students the proper way to handle the use of content that is not their own.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Warm Up <ul style="list-style-type: none"> <li>○ Write a character sketch</li> </ul> </li> <li>● Main Activity <ul style="list-style-type: none"> <li>○ Digital sharing</li> </ul> </li> <li>● Wrap Up <ul style="list-style-type: none"> <li>○ Reflection</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Course E Lesson 7: Digital Sharing</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Interpret ethical sharing of copyrighted material vs. sharing that is not ethical</li> <li>● Understand their own rights regarding materials that they have created</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.5</b>—Modify, remix, or incorporate portions of an existing program into one’s own work to develop something new or add more advanced features.</li> <li>● <b>AP.1B.5a</b>—Students should be able to modify and/or reuse portions of an existing program into their own work to create something new.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up <b>15 min</b></li> <li>● Main Activity <b>35 min</b></li> <li>● Wrap Up <b>15 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> <li>● <a href="#">Digital Sharing Lesson Plan</a></li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> <li>● <a href="#">Digital Sharing Ethics (Video)</a></li> </ul>
<p>Subject integrated:</p>	<p>Social Studies</p>
<p>Other standards addressed:</p>	<p><b>CI.4.3.1</b>—Examine responsibilities as citizens, such as obeying rules and laws.</p>
<p>Vocabulary:</p>	<p><u>Copyright</u>: The exclusive legal right to print, publish, perform, film, or record literary, artistic, or musical material, and to authorize others to do the same</p>
<p>Notes:</p>	

**Week 29: Code.org, Course E, Lesson 14—Conditionals in Minecraft Voyage Aquatic**

<p>Lesson overview:</p> 	<p><b>Purpose:</b> This set of puzzles will work to solidify and build on the knowledge of loops and introduce conditionals. By pairing these two concepts together, students will be able to explore the potential for creating fun and innovative programs in a new and exciting environment.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Key vocabulary</li> </ul> </li> <li>● Conditionals in Minecraft: Voyage Aquatic <ul style="list-style-type: none"> <li>○ Minecraft: Voyage Aquatic Introduction (video)</li> <li>○ Skill Building</li> <li>○ Minecraft: Voyage Aquatic Repeat Until (video)</li> <li>○ Skill Building</li> <li>○ Minecraft: Voyage Aquatic Conditionals (video)</li> <li>○ Skill building</li> <li>○ Minecraft: Voyage Aquatic Congratulations</li> <li>○ Free play</li> </ul> </li> <li>● Reflections</li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Course E Lesson 14: Conditionals in Minecraft Voyage Aquatic</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Define circumstances when certain parts of a program should run and when they should not</li> <li>● Determine whether a conditional is met based on criteria</li> </ul> <p>Standards</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.3</b>—Create programs that include sequences, events, loops, and conditionals.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up/Introduction <b>15 min</b></li> <li>● Main Activity <b>30 min</b></li> <li>● Wrap Up/Reflection <b>15 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
<p>Subject integrated:</p>	<p>Math</p>
<p>Other standards addressed:</p>	<p><b>4.NBT.5</b>—Multiply a whole number of up to four digits by a one-digit whole number, and multiply two two-digit numbers, using strategies based on place value and the properties of operations. Illustrate and explain the calculation by using equations, rectangular arrays, and/or area models.</p>
<p>Vocabulary:</p>	<p><u>Condition</u>: Something a program checks to see if it is true before allowing an action</p> <p><u>Conditionals</u>: Statements that only run under certain conditions</p>
<p>Notes:</p>	

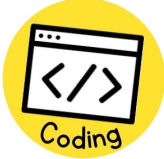
## Week 30: Weather Safety

<p>Lesson overview:</p>	<p><b>Purpose:</b></p>
-------------------------	------------------------

	<p>Students will create an animated story with a stormy setting to discuss weather safety.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Review weather terms</li> </ul> </li> <li>● Main Activity <ul style="list-style-type: none"> <li>○ Follow the <a href="#">Story Telling</a> using CS First. Students will follow along with the videos provided from CS First while working to create their own animated story using scratch (link is provided through CS First).</li> </ul> </li> <li>● Wrap Up <ul style="list-style-type: none"> <li>○ Students will share their stories with a partner</li> </ul> </li> <li>● Keyboarding Practice <ul style="list-style-type: none"> <li>○ If time remains, have students use resources from above to practice keyboarding.</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">Story Telling Lesson Plan</a></li> <li>● <a href="#">Story Telling</a> Using CS First, click on Storytelling</li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Students will create an animated story with a story setting to discuss weather safety using Scratch</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.10</b>—Create programs that include sequences, events, loops, and conditionals.</li> <li>● <b>AP.1B.11</b>—Decompose problems into smaller, manageable subproblems to facilitate the program development process.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up/Introduction <b>5 min</b></li> <li>● Main Activity <b>35 min</b></li> <li>● Wrap Up <b>10 min</b></li> <li>● Keyboarding Practice <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
<p>Subject integrated:</p>	<p>Science</p>
<p>Other standards addressed:</p>	<p><b>E.4.9C.5</b>—Obtain and communicate information about severe weather phenomena (e.g., thunderstorms, hurricanes, or tornadoes) to explain steps humans can take to reduce the impact of severe weather events</p>
<p>Vocabulary:</p>	<p><u>Condition</u>: Something a program checks to see if it is true before allowing an action</p> <p><u>Conditionals</u>: Statements that only run under certain conditions</p> <p><u>While loop</u>: A loop that continues to repeat while a condition is true</p>
<p>Notes:</p>	

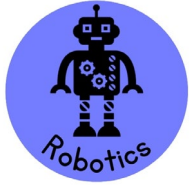
<b><u>Week 31: Famous Mississippians</u></b>	
Lesson overview:	<u>Purpose:</u>



	<p>Students will choose a famous Mississippian and write a story about their accomplishments. They will make this story come to life by utilizing Scratch.</p> <p><u>Lesson:</u></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Project expectations</li> <li>○ Choose your famous Mississippian</li> </ul> </li> <li>● Main Activity <ul style="list-style-type: none"> <li>○ Create animation</li> </ul> </li> <li>● Wrap Up and Share</li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Scratch: Famous Mississippians</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Create an animation</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.3a</b>—Students should be able to create programs that include sequences, events, loops, and conditionals.</li> <li>● <b>AP.1B.6b</b>—Students should document the plan as, for example, a storyboard, flowchart, pseudocode, or story map.</li> </ul>
<p>Time needed:</p>	<p><b><u>Total Time:</u> 60 min</b></p> <ul style="list-style-type: none"> <li>● Introduction <b>10 min</b></li> <li>● Main Activity <b>40 min</b></li> <li>● Wrap Up <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
<p>Subject integrated:</p>	<p>Social Studies</p>
<p>Other standards addressed:</p>	<p><b>CR.4.1.1</b>—Analyze the Civil Rights Movement to determine the social, political, and economic impact on Mississippi.</p>
<p>Vocabulary</p>	<p><u>Condition:</u> Something a program checks to see if it is true before allowing an action</p> <p><u>Conditionals:</u> Statements that only run under certain conditions</p> <p><u>While Loop:</u> A loop that continues to repeat while a condition is true</p>
<p>Notes:</p>	

## **Week 32: Points, Lines, Segments, Rays, and Angles**

<p>Lesson overview:</p>	<p><b><u>Purpose:</u></b></p>
-------------------------	-------------------------------



**\*Robotics Lesson with Edison Robotics Robot\***  
*If you do not have access to a robot (code & go mouse, Edison, Sphero Sprk+, etc.) a person/persons can "be" the robot*  
 The student will remix a scratch lesson to review math lines, segments, rays, and angles.

- Lesson:**
- Introduction
    - Review vocabulary terms for math
  - Main Activity
    - In various groups, the student will remix [Line Tracking Lesson](#) to include different types of lines, such as (but not limited to) one set of parallel lines, one set of perpendicular lines, and two intersecting lines that are not perpendicular.
    - The student will then plug in their Edison Robot and upload their created maze to the Edison.
  - Wrap Up
    - All groups will collaboratively reflect on the similarities and differences of the mazes created as well as the use of the various types of lines.

Lesson links/resources:	<a href="#">Scratch: Line Tracking</a>
CS standards addressed:	Students will be able to: <ul style="list-style-type: none"> <li>● Remix a scratch lesson by adding different types of lines</li> <li>● Upload created mazes to Edison</li> </ul> Standards: <ul style="list-style-type: none"> <li>● <b>AP.1B.3</b>—Create programs that include sequences, events, loops, and conditionals.</li> </ul>
Time needed:	<b>Total Time: 60 min</b> <ul style="list-style-type: none"> <li>● Introduction <b>10 min</b></li> <li>● Main Activity <b>40 min</b></li> <li>● Wrap Up <b>10 min</b></li> </ul>
Materials needed:	Teachers: <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> <li>● Codable Robot (Some examples include: <a href="#">Edison robots</a>, <a href="#">Code and Go Mouse</a>, <a href="#">Botley</a>, <a href="#">Dash</a>)</li> </ul> Students: <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
Subject integrated:	Math
Other standards addressed:	<b>4.G.1</b> —Draw points, lines, line segments, rays, angles (right, acute, obtuse), and perpendicular and parallel lines. Identify these in two-dimensional figures.
Vocabulary:	<b>Program:</b> An algorithm that has been coded into something that can be run by a machine <b>Programming:</b> The art of creating a program

Notes:

**Week 33: Digital Sharing Animation**

Lesson overview:

**Purpose:**



Students will apply their understanding of sharing personal and private information on the web by creating an interactive poster in this mini project, after reading and discussing 2 passages about digital sharing. The students will compare and contrast the information about the same topic in an animation.

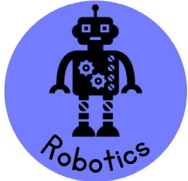
**Lesson:**

- Introduction
  - Recall information from previous digital sharing lessons about sharing personal and private information
- Main Activity
  - Pre-writing-brainstorming-Venn diagram
  - Create animation
- Wrap Up
  - Present interactive poster

Lesson links/resources:	<a href="#">Scratch: Digital Sharing Animation</a>
CS standards addressed:	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Use public domain and common media, and refrain from copying and using material created by others</li> <li>● Discuss cybersecurity issues and protection of personal information</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>1B.IC.21</b>—Use public domain or creative commons media and refrain from copying or using material created by others without permission.</li> <li>● <b>1B.NI.05</b>—Discuss real-world cybersecurity problems and how personal information can be protected.</li> </ul>
Time needed:	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Introduction <b>10 min</b></li> <li>● Main Activity <b>40 min</b></li> <li>● Wrap Up <b>10 min</b></li> </ul>
Materials needed:	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
Subject integrated:	ELA
Other standards addressed:	<b>RI.4.9</b> —Integrate information from two texts about the same subject knowledgeably.
Vocabulary:	<u>Copyright</u> : The exclusive legal right to print, publish, perform, film, or record literary, artistic, or musical material, and to authorize others to do the same
Notes:	

**Week 34: Maze Mat Race**

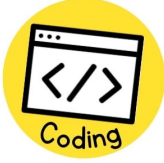
Lesson overview:	<i>Robotics Lesson with Code &amp; Go Mouse:</i>
------------------	--

	<p><u>Purpose:</u> Reading comprehension lesson for students to strengthen their skills in referring to details, theme, character, setting, or word meaning, using interactive Maze Mat Race game.</p> <p><u>Lesson:</u></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Discuss game rules and concept</li> </ul> </li> <li>● Main Activity <ul style="list-style-type: none"> <li>○ Divide the class into teams</li> <li>○ Begin game</li> </ul> </li> <li>● Wrap Up <ul style="list-style-type: none"> <li>○ Debugging</li> <li>○ Reflection</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">Maze Mat Race Lesson Plan</a></li> <li>● <a href="http://www.kahoot.com">www.kahoot.com</a></li> <li>● <a href="http://www.quizziz.com">www.quizziz.com</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Code a robot to maneuver through a maze</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.1</b>—Compare and refine multiple algorithms for the same task and determine which is the most appropriate.</li> <li>● <b>AP.1B.4</b>—Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Introduction <b>10 min</b></li> <li>● Main Activity <b>40 min</b></li> <li>● Wrap Up <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> <li>● Robot (Some examples include: <a href="#">Code and Go Mouse</a>, <a href="#">Botley</a>, <a href="#">Dash</a>)</li> <li>● Coding maze (Should be already coded before the game)</li> <li>● Grade level passage (Can be passage previously discussed)</li> <li>● Access to Kahoot or Quizizz</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
<p>Subject integrated:</p>	<p>ELA</p>
<p>Other standards addressed:</p>	<ul style="list-style-type: none"> <li>● <b>RL.4.1</b>—Refer to details and examples in a text when explaining what the text says explicitly and when drawing inferences from the text.</li> <li>● <b>RL.4.2</b>—Determine a theme of a story, drama, or poem from details in the text; summarize the text.</li> <li>● <b>RL.4.3</b>—Describe in depth a character, setting, or event in a story or drama, drawing on specific details in the text (e.g., a character's thoughts, words, or actions).</li> <li>● <b>RL.4.4</b>—Determine the meaning of words and phrases as they are used in a text, including those that allude to significant characters found in mythology (e.g., Herculean).</li> </ul>
<p>Vocabulary:</p>	<p><u>Program:</u> An algorithm that has been coded into something that can be</p>

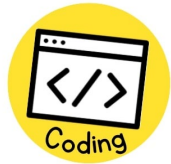
	run by a machine <u>Programming</u> : The art of creating a program
Notes:	If the teacher does not have a Code & Go Mouse, a student could act as the mouse to maneuver through the maze.

**Week 35: Energy Sources**

Lesson overview:	<u>Purpose</u> :
------------------	------------------

	<p>Students will create a presentation in support of the energy source that they feel is most sustainable and environmentally friendly.</p> <p><u>Lesson:</u></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Interactive presentation</li> </ul> </li> <li>● Main Activity <ul style="list-style-type: none"> <li>○ Build Your presentation using "Pitch your Passion"</li> <li>○ Survey-interaction presentation</li> </ul> </li> <li>● Wrap Up</li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">CS First: Energy Sources</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Express what they feel is the most sustainable and environment-friendly energy source through the creation of an animation</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.5</b>—Modify, remix, or incorporate portions of an existing program into one's own work to develop something new or add more advanced features.</li> <li>● <b>AP.1B.6</b>—Use an iterative process to plan the development of a program by including others' perspectives and considering user preferences.</li> </ul>
<p>Time needed:</p>	<p><b><u>Total Time: 60 min</u></b></p> <ul style="list-style-type: none"> <li>● Warm Up/Introduction <b>10 min</b></li> <li>● Main Activity <b>35 min</b></li> <li>● Wrap Up <b>15 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
<p>Subject integrated:</p>	<p>Science</p>
<p>Other standards Addressed:</p>	<ul style="list-style-type: none"> <li>● <b>E.4.10.1</b>—Organize simple data sets to compare energy and pollution output of various traditional, nonrenewable resources (e.g., coal, crude oil, wood).</li> <li>● <b>E.4.10.2</b>—Use technology or informational text to investigate, evaluate, and communicate various forms of clean energy generation</li> </ul>
<p>Vocabulary:</p>	<p><u>Parallelism</u>: The process of events happening at the same time, either independently or interdependently</p> <p><u>Debugging</u>: The process of identifying and fixing error(s) in a program when it is not functioning as expected</p> <p><u>Control structures</u>: Sections of code that order the direction or flow of how a program functions. The control structure in this lesson focuses on loops</p>
<p>Notes:</p>	

<p><b><u>Week 36: Code.org, Course E, Lesson 15—Conditionals With the Farmer</u></b></p>	
<p>Lesson overview:</p>	<p><b><u>Purpose:</u></b></p>



This lesson introduces students to while loops and "if/else" statements.

**Lesson:**

- Introduction
  - Key vocabulary
- Conditionals With the Farmer
  - Skill building
  - While loops with the farmer (video)
  - Prediction
  - Skill building
  - Repeat until blocks (video)
  - Skill building
  - "If/else" (video)
  - Challenge
  - Practice
- Wrap Up/Reflection

Lesson links/resources:

[Course E Lesson 15: Conditionals With Farmer](#)

CS standards addressed:

- Students will be able to:
- Define circumstances when certain parts of a program should run and when they should not
  - Determine whether a conditional is met based on criteria
- Standards
- **AP.1B.3**—Create programs that include sequences, events, loops, and conditionals [CONTROL] (P5.2).
  - **AP.1B.3a**—Students should be able to create programs that include sequences, events, loops and conditionals.
  - **AP1.B.4**—Decompose (break down) problems into smaller, manageable subproblems to facilitate the program.
  - **AP1.B.4a**—Students should be able to break down problems into smaller, simpler tasks.

Time needed:

- Total Time: 60 min**
- Warm Up **15 min**
  - Main Activity **30 min**
  - Wrap Up/Reflection **15 min**

Materials needed:

- Teachers:
- Smartboard/projector with sound
  - [Pause and Think Online](#) - Video
- Students:
- Student devices with access to the internet

Subject integrated:

Social Studies

Other standards addressed:

- **G.4.1.1**—Compare and contrast the ten geographical regions of Mississippi in terms of soil, landforms, etc.
- **G.4.2.3**—Describe the opportunity cost of choices made within Mississippi (e.g., cotton farming vs. soybean farming, pastureland vs. industrial development, beaches vs. casinos, landfills vs. parks, etc.).

Vocabulary:

Condition: Something a program checks to see if it is true before allowing an action  
Conditionals: Statements that only run under certain conditions  
While Loop: A loop that continues to repeat while a condition is true

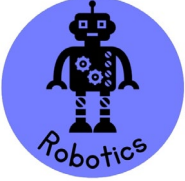


Notes:

**Week 37: Angle Measures**

Lesson overview:

**Purpose:**

	<p>Students will answer questions about angles then program a robotic mouse to run through a maze.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Review terms</li> </ul> </li> <li>● Main Activity <ul style="list-style-type: none"> <li>○ Answer questions while playing Kahoot</li> <li>○ Teams that answer the question correctly, get to enter a code on their robot. If it is a high-level question, it can be worth more.</li> </ul> </li> <li>● Wrap Up <ul style="list-style-type: none"> <li>○ Debugging</li> <li>○ Reflection</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">Kahoot</a></li> <li>● <a href="#">Angle Measures Lesson Plan</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Program a robotic mouse to move through a maze while answering questions</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.4</b>—Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Introduction <b>10 min</b></li> <li>● Main Activity <b>40 min</b></li> <li>● Wrap Up <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> <li>● Robot (Some examples include: <a href="#">Code and Go Mouse</a>, <a href="#">Botley</a>, <a href="#">Dash</a>)</li> <li>● Coding maze (Should be already coded before the game)</li> <li>● Grade level passage (Can be passage previously discussed)</li> <li>● Access to Kahoot or Quizizz</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
<p>Subject integrated:</p>	<p>Math</p>
<p>Other standards addressed:</p>	<p><b>4.MD.7</b>—Recognize angle measure as additive. When an angle is decomposed into non overlapping parts, the angle measure of the whole is the sum of the angle measures of the parts. Solve addition and subtraction problems to find unknown angles on a diagram in real world and mathematical problems, e.g., by using an equation with a symbol for the unknown angle measure</p>
<p>Vocabulary:</p>	<p><u>Condition</u>: Something a program checks to see if it is true before allowing an action</p> <p><u>Conditionals</u>: Statements that only run under certain conditions</p> <p><u>While Loop</u>: A loop that continues to repeat while a condition is true</p>

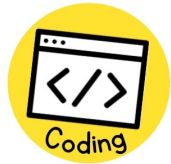
Notes:

### **Week 38: Text Structures (Coding)**

Lesson overview:

**Purpose:**

Today, students will create animated characters that will discuss text structures. The students will use Scratch to discuss text structures.



**Lesson:**

- Introduction
  - The teacher will review using Scratch with students.
- Main Activity
  - The students will create different animations where the characters are discussing the different text structures.
- Wrap Up
  - The students can share their animations with the group
  - Reflection

Lesson links/resources:

[Scratch: Text Structures \(Coding\)](#)

CS standards addressed:

Students will be able to:

- Define circumstances when certain parts of a program should run and when they should not
- Determine whether a conditional is met based on criteria

Standards:

- **AP.1B.4**—Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process

Time needed:

**Total Time: 60 min**

- Introduction **10 min**
- Main Activity **40 min**
- Wrap Up **10 min**

Materials needed:

Teachers:

- Smartboard/projector with sound

Students:

- Student devices with access to the internet

Subject integrated:

ELA

Other standards addressed:

**RI.4.5**—Determine the overall structure (chronology, comparison, cause/effect, etc.) of events, ideas, or information in a text or pair of text.

Vocabulary:

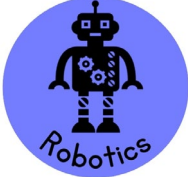
Notes:

**Week 39: Text Structure (Robotic)**

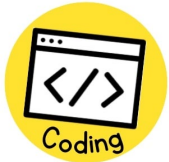
Lesson overview:

**Purpose:**

Students will answer questions about Angles then program a robotic mouse to run through a maze. The teacher will need to pull questions on text features for this activity.

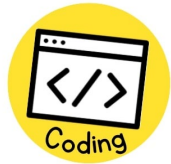
	<p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Review text structures</li> </ul> </li> <li>● Main Activity <ul style="list-style-type: none"> <li>○ Answer questions while playing Kahoot</li> <li>○ Teams that answer the question correctly get to enter a code on their robot. If it is a high-level question, it can be worth more.</li> </ul> </li> <li>● Wrap Up <ul style="list-style-type: none"> <li>○ Debugging</li> <li>○ Reflection</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">Kahoot on Text Structure</a></li> <li>● <a href="#">Text Structure (Robotic) Lesson Plan</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Program a robotic mouse to work through a maze</li> </ul> <p>Standards</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.1</b>—Compare and refine multiple algorithms for the same task and determine which is the most appropriate.</li> <li>● <b>AP.1B.4</b>—Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min:</b></p> <ul style="list-style-type: none"> <li>● Introduction <b>10 min</b></li> <li>● Main Activity <b>40 min</b></li> <li>● Wrap Up <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> <li>● Robot (Some examples include: <a href="#">Code and Go Mouse</a>, <a href="#">Botley</a>, <a href="#">Dash</a>)</li> <li>● Maze mat</li> <li>● Coding maze (Should be already coded before the game)</li> <li>● Grade level passage: (Can be passage previously discussed)</li> <li>● Access to Kahoot or Quizizz</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
<p>Subject integrated:</p>	<p>ELA</p>
<p>Other standards addressed:</p>	<p><b>RI.4.5</b>—Describe the overall structure (e.g., chronology, comparison, cause/effect, problem/solution) of events, ideas, concepts, or information in a text or part of a text.</p>
<p>Vocabulary:</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:

<p>Lesson overview:</p> 	<p><b>Purpose:</b> This lesson is meant to further push students to use conditionals with functions. These puzzles are intended to increase problem solving and critical thinking skills.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Discuss advantages and disadvantages of using functions in a program</li> </ul> </li> <li>● Online Puzzles <ul style="list-style-type: none"> <li>○ The Harvester (video)</li> <li>○ Skill building</li> <li>○ Challenge</li> <li>○ Practice</li> <li>○ Prediction</li> </ul> </li> <li>● Reflection/Share</li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Course E Lesson 16: Functions With Harvester</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Recognize when a function could help to simplify a program</li> <li>● Use predetermined functions to complete commonly repeated tasks</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1B.1</b>—Compare and refine multiple algorithms for the same task and determine which is the most appropriate.</li> <li>● <b>AP.1B.4</b>—Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up/Introduction <b>10 min</b></li> <li>● Main Activity <b>30 min</b></li> <li>● Wrap Up/Reflection/Share <b>20 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teachers:</p> <ul style="list-style-type: none"> <li>● Smartboard/projector with sound</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Student devices with access to the internet</li> </ul>
<p>Subject integrated:</p>	<p>Math</p>
<p>Other standards addressed:</p>	<p><b>4.OA.5</b>—Generate a number or shape pattern that follows a given rule. Identify apparent features of the pattern that were not explicit in the rule itself.</p>
<p>Vocabulary:</p>	<p><u>Conditionals</u>: Statements that only run under certain conditions <u>Function</u>: A piece of code that you can easily call over and over again</p>
<p>Notes:</p>	

## Week 41: Code.org, Course E, Lesson 17—Designing for Accessibility

<p>Lesson overview:</p>	<p><b>Purpose:</b></p>
-------------------------	------------------------



Through learning about accessibility, students recognize the impacts of computing beyond their own lives. Accessibility might not seem like a relevant CS topic but creating technology that is accessible for underserved users helps make tech better for everyone else as well.

**Lesson:**

- Introduction
  - Context-setting
- Designing for Accessibility
  - Scenarios
  - App redesign
- Reflection
- Keyboarding Practice
  - If time remains, have students use resources from above to practice keyboarding.

Lesson links/resources:

[Course E Lesson 17: Designing for Accessibility](#)

CS standards addressed:

Student will be able to:

- Describe the impact of mobile apps on the modern world
- Explain why accessibility is an important part of designing an app for users
- Improve upon an existing app design by addressing the accessibility needs of users

Standards

- **AP.1B.5**—Modify, remix, or incorporate portions of an existing program into one's own work to develop something new or add more advanced features.
- **AP.1B.6**—Use an iterative process to plan the development of a program by including others' perspectives and considering user preferences.
- **CS.1B.3**—Determine potential solutions to solve simple hardware and software problems using common troubleshooting strategies
- **IC.1B.1**—Discuss computing technologies that have changed the world and express how those technologies influence and are influenced by cultural practices.
- **IC.1B.2**—Brainstorm ways to improve the accessibility and usability of technology products for the diverse needs and wants of users.
- **IC.1B.3**—Seek diverse perspectives for the purpose of improving computational artifacts.

Time needed:

**Total Time: 60 min**

- Warm Up/Introduction **5 min**
- Main Activity **35 min**
- Wrap Up **5 min**
- Keyboarding Practice **15 min**

Materials needed:

Teachers:

- Smartboard/projector with sound

Students:

- Pencil
- Paper
- Art supplies
- [Types of Disabilities](#) - Resource for the students
- Student devices with access to the internet
- [Designing for Accessibility](#)-Slides



Subject integrated:	Writing
Other standards addressed:	<b>W.4.6</b> —With some guidance and support from adults, use technology, including the Internet, to produce and publish writing as well as to interact and collaborate with others; demonstrate sufficient command of keyboarding skills to type a minimum of one page in a single sitting.
Vocabulary:	None
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**

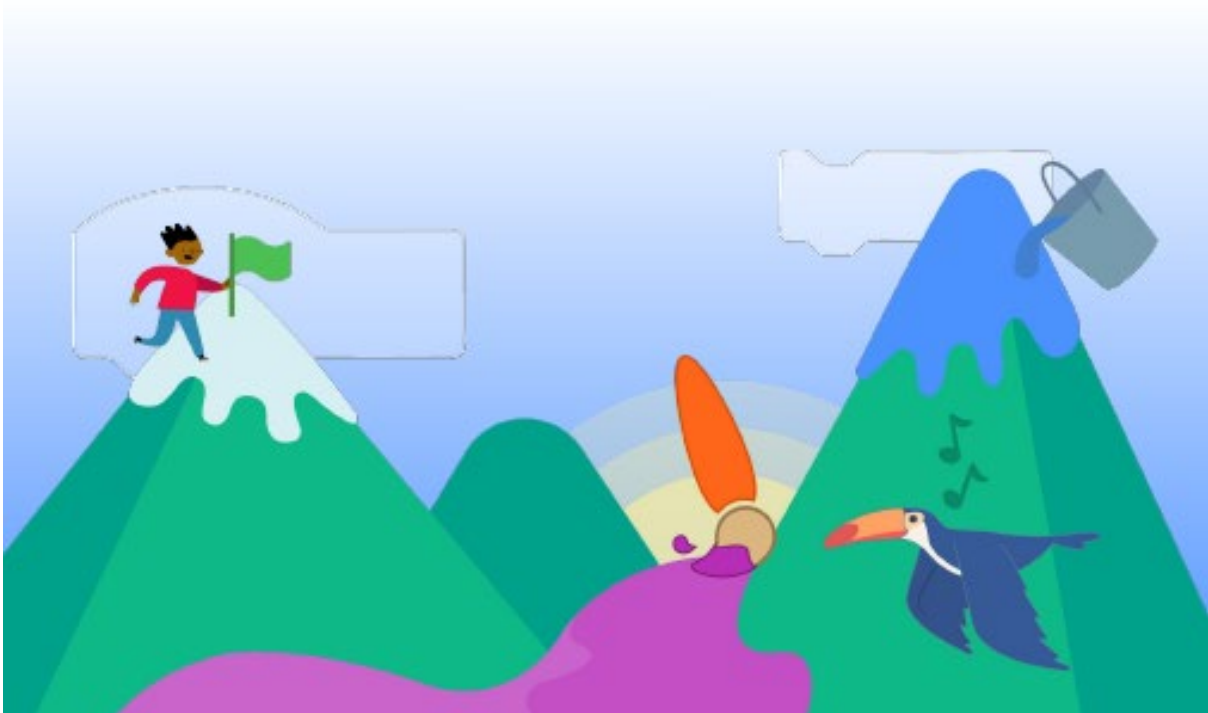
<https://support.code.org/hc/en-us>

## 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

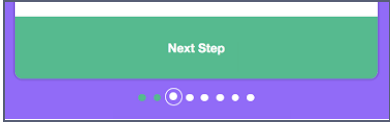


Next Step

#### 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

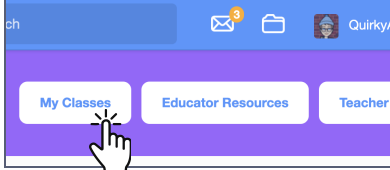

**Be sure to make a note of your username and password.**



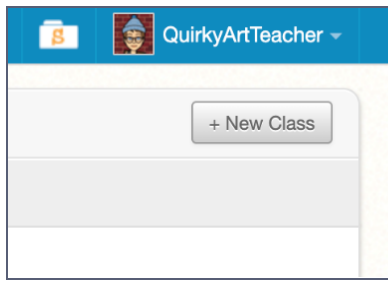
	<p>Click through each step to <b>complete registration</b>.</p>
	<p>Log into your email and confirm your email address.</p> <p>Check your spam folder if you do not see the email.</p> <p>Once you have <b>confirmed your email address</b>, we'll review your account.</p>
	<p>Once your account has been reviewed and approved, <b>you will receive a welcome email</b>. Then, you can <b>log into your teacher account at <a href="https://scratch.mit.edu">scratch.mit.edu</a></b>!</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 “<b>My Classes</b>.”</p>
	<p>You can also access your classes from the dropdown under your username.</p>



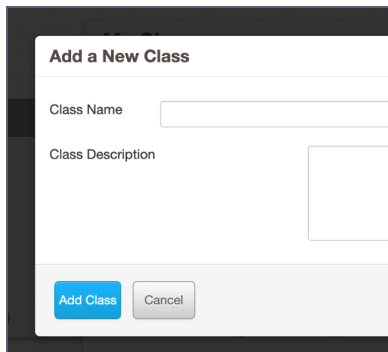


To create a class, click the “+ **New Class**” button at the top right of the page.

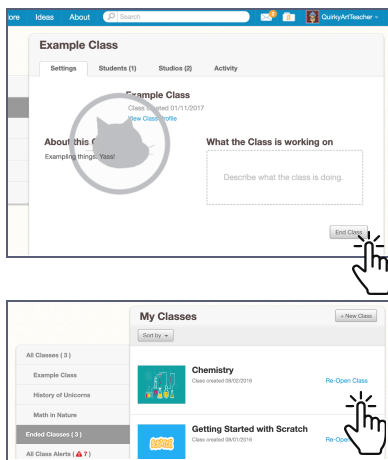
Enter the class name and description.

**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.

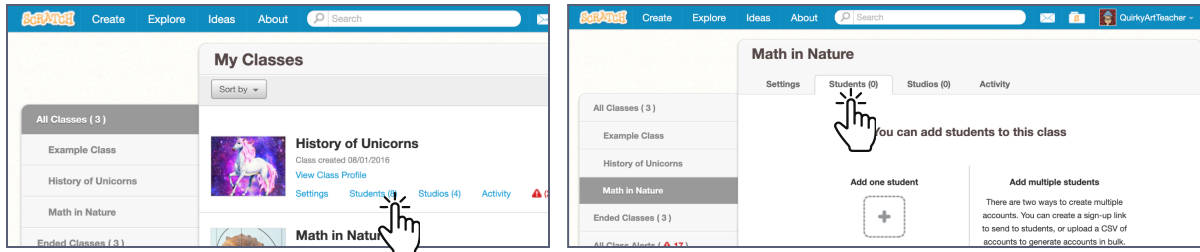
When you end a class, your class profile page will be hidden and your students will no longer be able to log in (but their projects and the class studios will still be visible on the site).

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

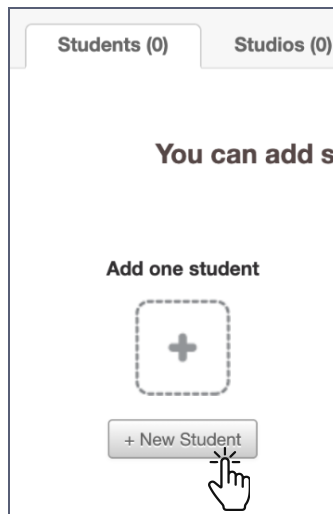
While on “My Classes,” select the class and then click on “Students” (either the link under the class name or the Students tab). Once created, your student accounts will appear here.



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** [X]

Add to Class: Math in Nature

Username: type student username here

⚠️ **Names 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.

Settings Students (0) Studios (0) Activity

**Sign-up Link** [X]

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”.

⚠️ **Names 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.

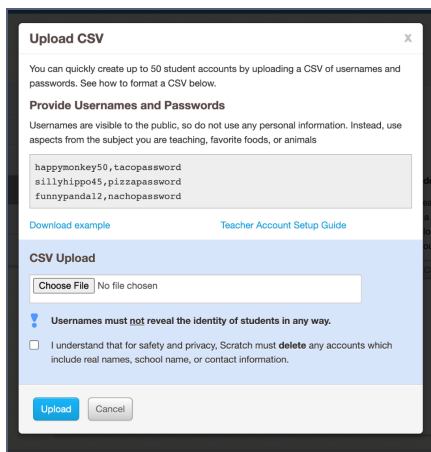
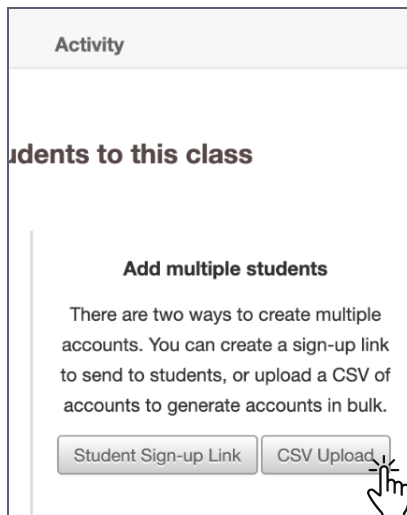
[Get Link]

[Close]





## Method 3: CSV Upload



The screenshot shows a spreadsheet template for student accounts. It has two columns labeled 'A' and 'B', and two rows of student data.

	A	B
1	student1	password1
2	student2	password2

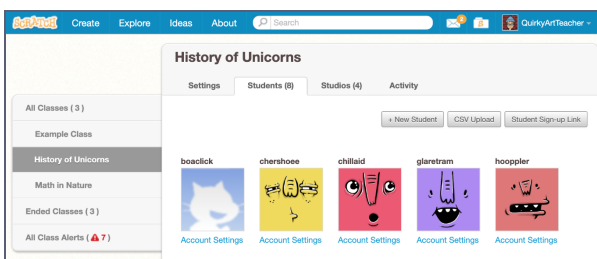
Click the “CSV Upload” button on the class page.

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

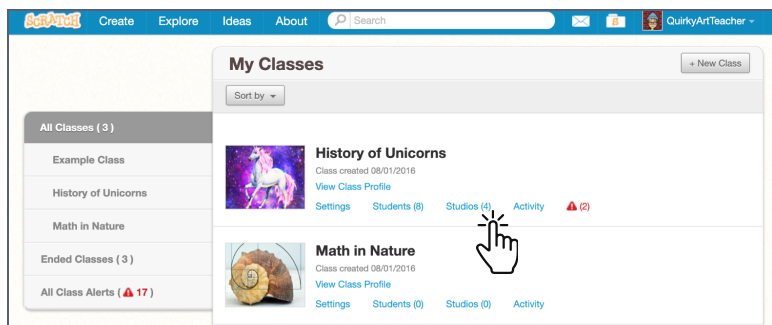
Studios allow you to create collections of student projects for specific classes or assignments. This makes it easier for you to view their projects throughout their creative process. It also makes it easier for students to collaborate and be inspired by each other's work.

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).

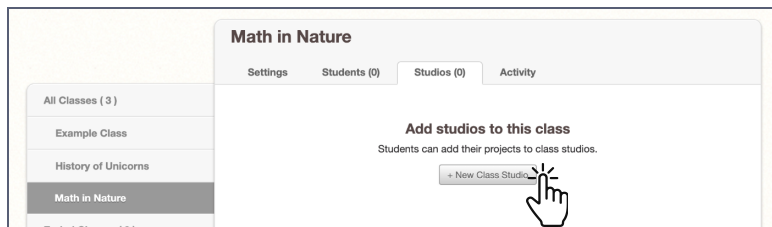
There are two ways to create a studio on a teacher account. Method one creates studios that automatically add all students in a class as curators. Method two creates studios without automatically adding students as curators, and students or any Scratcher can be individually added as curators.

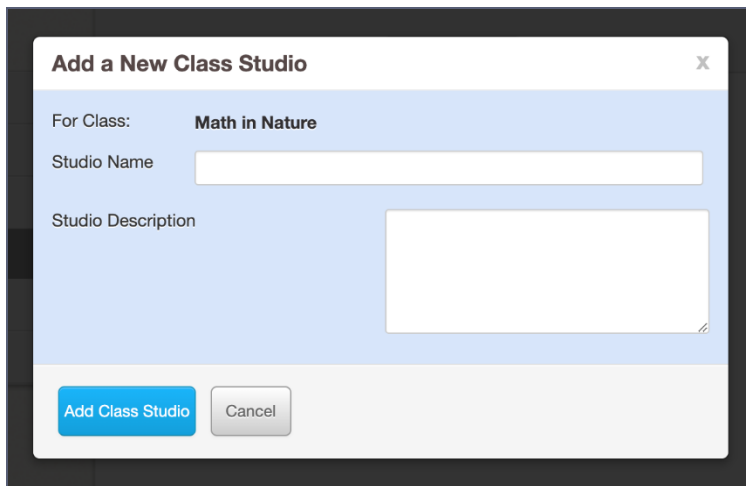
## Method 1: Create a studio that automatically adds all students in a class as curators



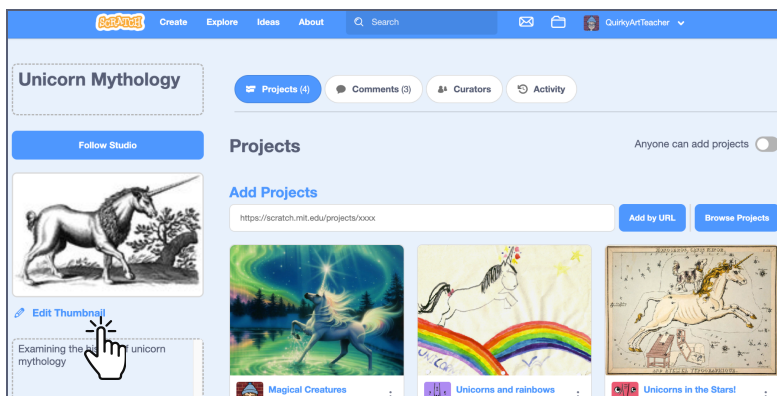
Once logged into your Scratch account, go to "My Classes."

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).

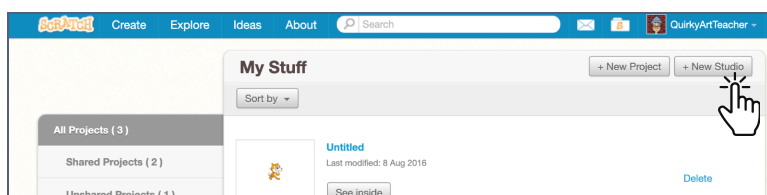


Then, click the “Add Class Studio” button.

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.

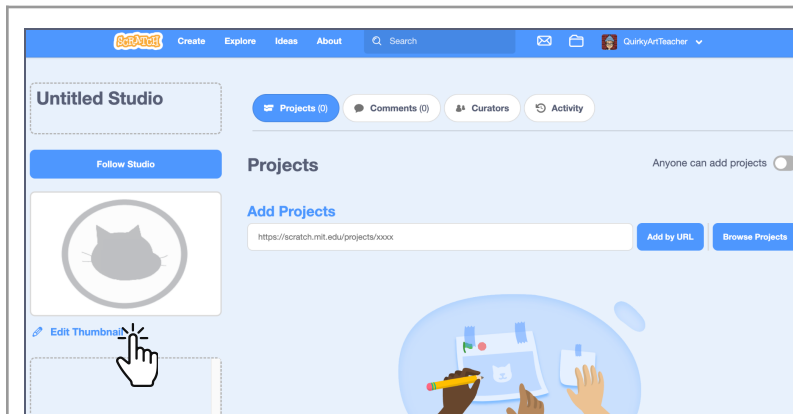
## Method 2: Create a studio without automatically adding students as curators



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.

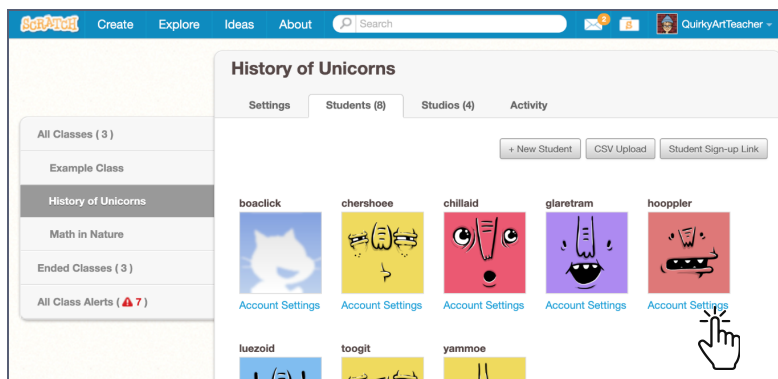
See our [Studio Guide](#) for detailed information on:

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

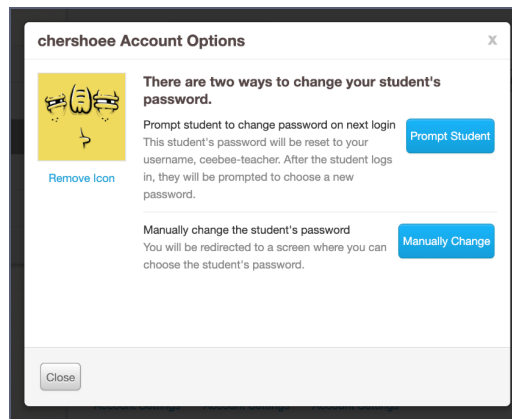


# Managing Your Students

## 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.



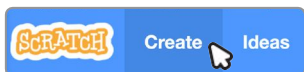


# GETTING STARTED

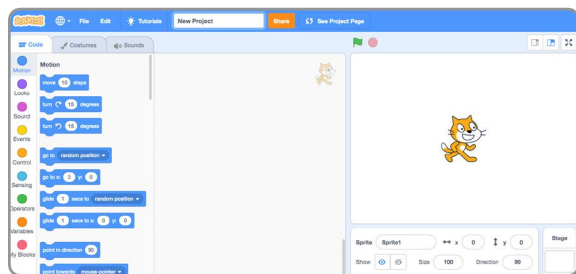
You can use Scratch online at: [scratch.mit.edu](https://scratch.mit.edu)

[scratch.mit.edu](https://scratch.mit.edu)

Once you've navigated to [scratch.mit.edu](https://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

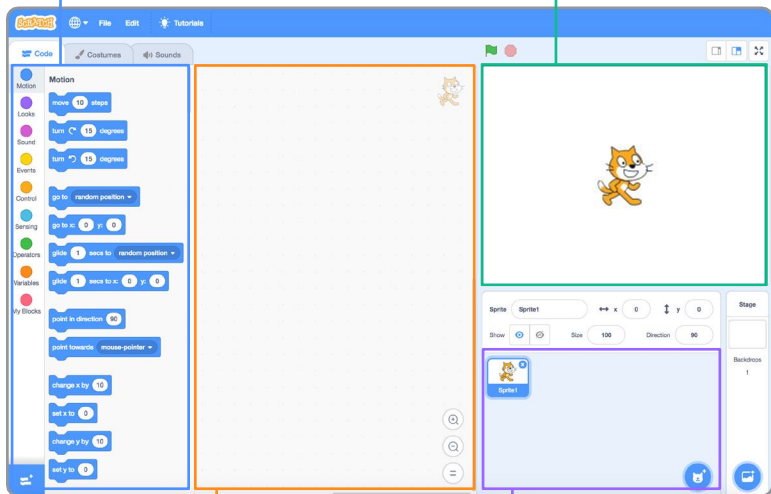
The Scratch Editor is where you create projects in Scratch. Here are its main parts:

## 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

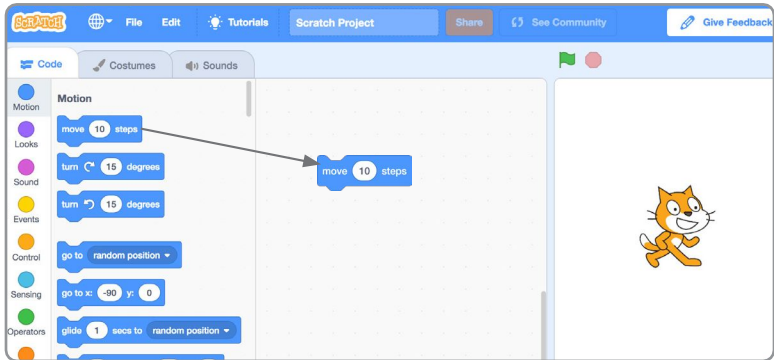
Click the thumbnail of a sprite to select it



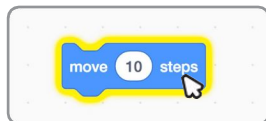


# LET'S CODE!

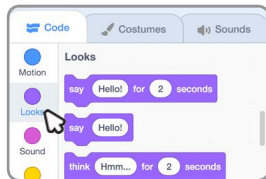
To code projects in Scratch, you snap together blocks. Start by dragging out a **move** block.



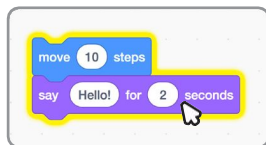
Click the block to try it.  
Does your cat move?



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.

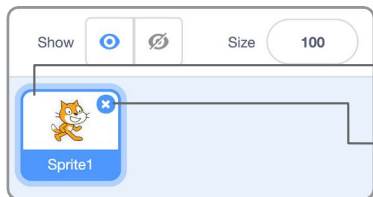
Draw your own sprite.



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.



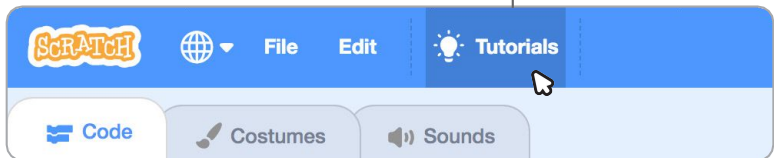
When you move your sprite, you can see its **x** and **y** position change.



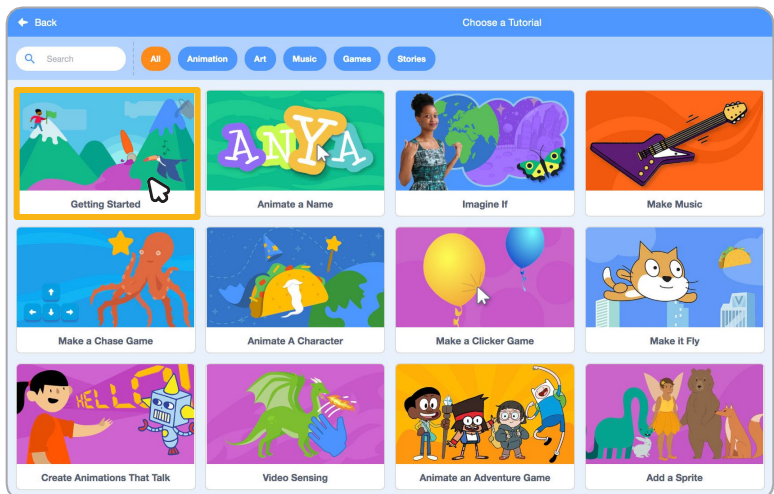
# TUTORIALS

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.

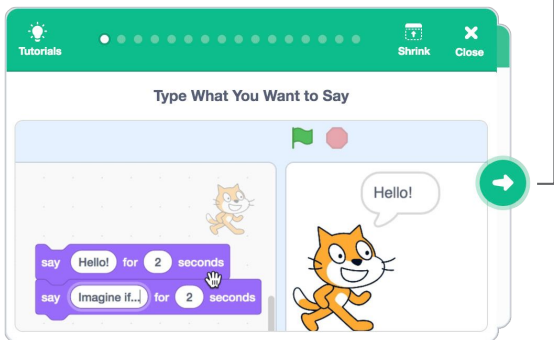




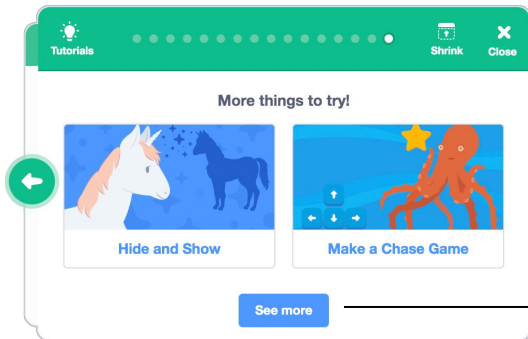
# TUTORIALS

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.

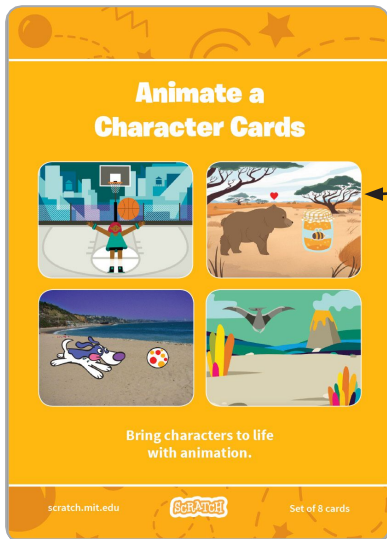


# CODING CARDS

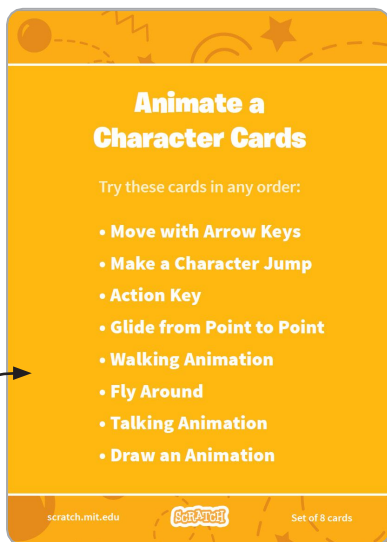
The Scratch **Coding Cards** provide another way to learn to create projects with Scratch. Download the cards at [scratch.mit.edu/ideas](https://scratch.mit.edu/ideas).

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!**

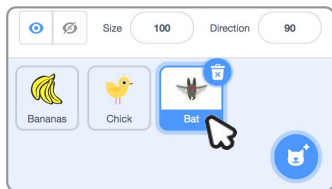




## GET CREATIVE WITH SPRITES!

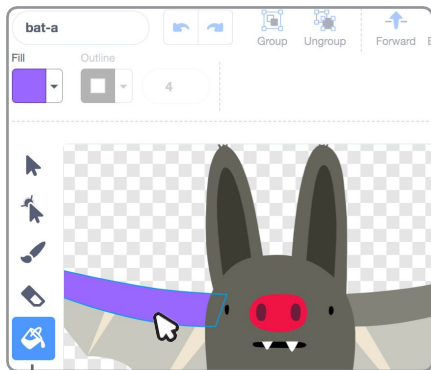
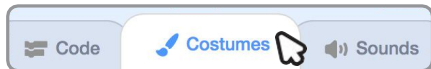
Scratch has its own paint tools, which allow you to customize sprites from the library, or even create sprites of your own.

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.

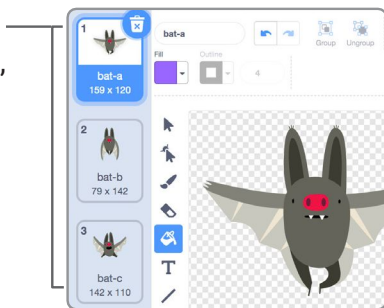
You can use the **paint bucket** tool to recolor different parts of a sprite.



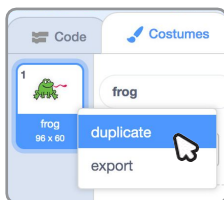
# GET CREATIVE WITH SPRITES!

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.

when this sprite clicked

next costume

wait 0.5 seconds

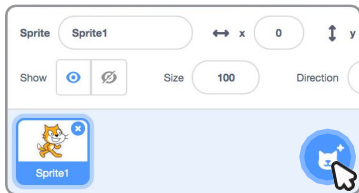
next costume



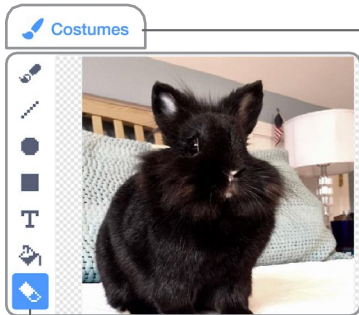
# ADD YOUR OWN PHOTOS

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.

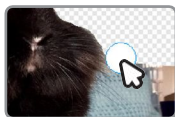


Hover over the New Sprite button, then select **Upload Sprite**.



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.



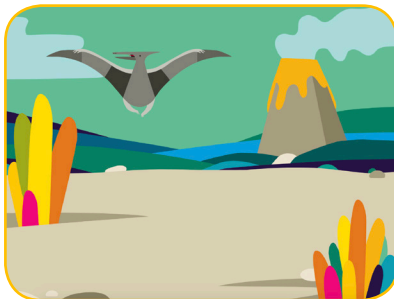
**Created by the Scratch Team** ([scratch.mit.edu](https://scratch.mit.edu)) and shared under the Creative Commons Attribution-ShareAlike 4.0 International Public License (CCbySA 4.0).

||| ||| ||| ||| ||| 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*

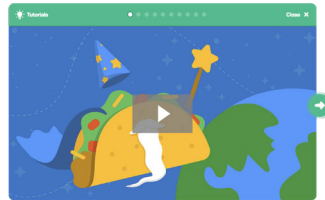
At the end of the session, gather together to share and reflect.

## 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](https://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](https://scratch.mit.edu/ideas)



### Make sure students sign into their Scratch accounts

Have students sign into their own Scratch accounts at [scratch.mit.edu](https://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



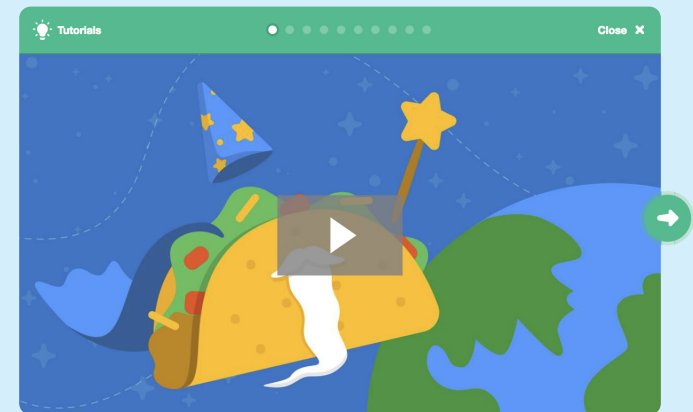
Begin by gathering the students to introduce the theme and spark ideas for projects.

### 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](https://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

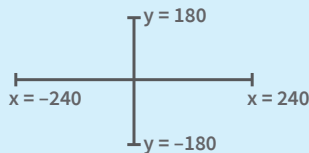
Choose left arrow from the menu.

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

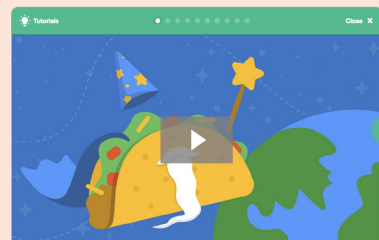
Ask students questions to get started

*Which character would you like to animate?*

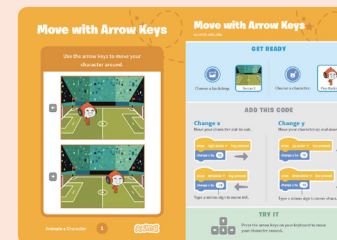
*What do you want your character to do?*

### Provide Resources

Offer options for getting started



Some students may want to follow the online tutorial: [scratch.mit.edu/animate](https://scratch.mit.edu/animate)



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



### 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

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](https://scratch.mit.edu/ideas)

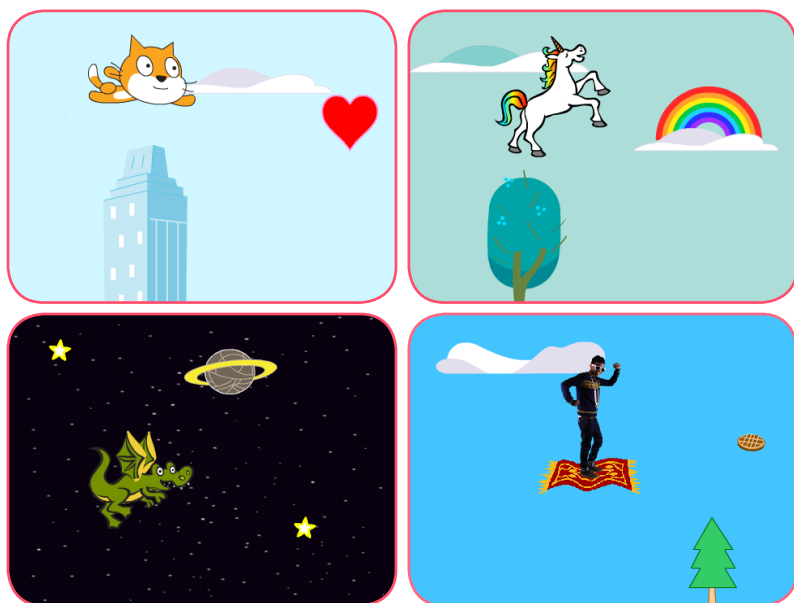
Scratch is a project of the Lifelong Kindergarten Group at the MIT Media Lab.



## 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*

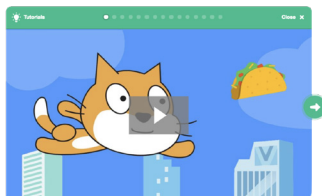
At the end of the session, gather together to share and reflect.

## Get Ready for the Lesson

Use this checklist to prepare for the lesson.

### Preview the Tutorial

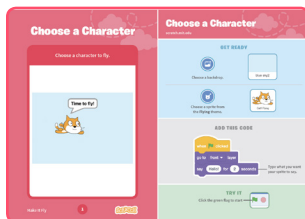
The *Make It Fly* tutorial shows students how to create their own projects. Preview the tutorial before your lesson and try the first few steps: [scratch.mit.edu/fly](https://scratch.mit.edu/fly)



### 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](https://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](https://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



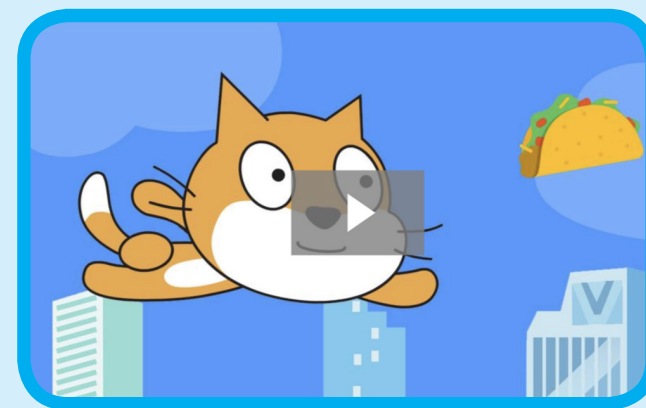
Begin by gathering the students to introduce the theme and spark ideas for projects.

### 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](https://scratch.mit.edu/fly) or [vimeo.com/llk/fly](https://vimeo.com/llk/fly)

## Demonstrate the First Steps



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

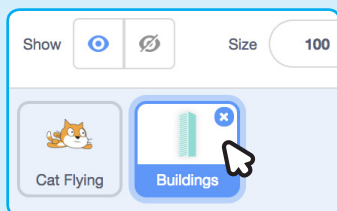
In Scratch, click Create.  
Choose a flying sprite from the library:



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

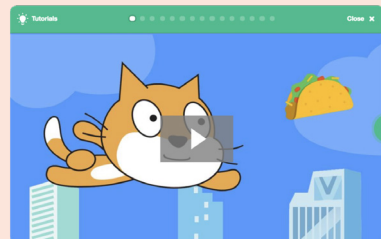
Ask students questions to get started

*What character would you like to make fly?*

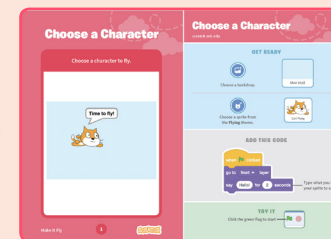
*Where will your character go flying?*

### Provide Resources

Offer options for getting started



Some students may want to follow the online tutorial:  
[scratch.mit.edu/fly](https://scratch.mit.edu/fly)



Others may want to explore using the activity cards:  
[scratch.mit.edu/fly/cards](https://scratch.mit.edu/fly/cards)

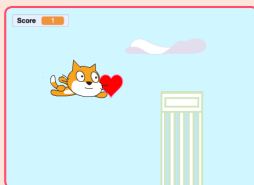
### Suggest Ideas for Starting

- 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

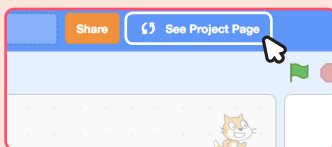
Here are some strategies to suggest to help students fix any bugs or difficulties they encounter:

- 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.



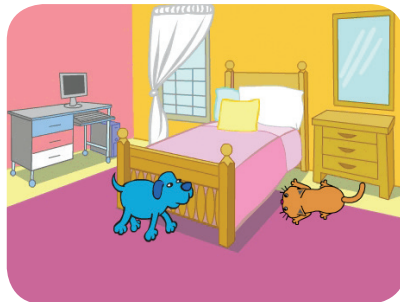
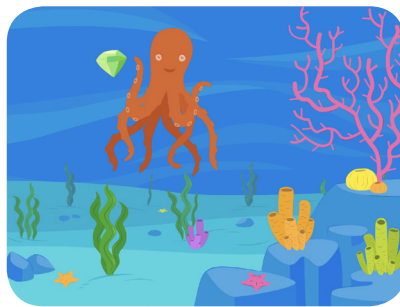
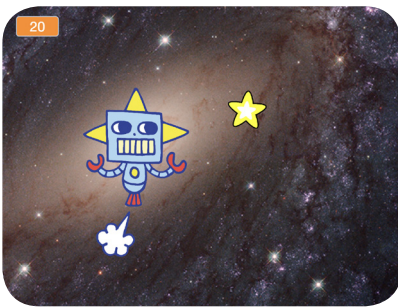
### Flying Stories

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*

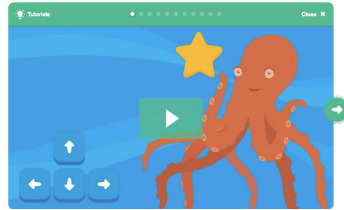
At the end of the session, gather together to share and reflect.

## Get Ready for the Lesson

Use this checklist to prepare for the lesson.

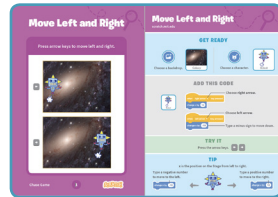
### Preview the Tutorial

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](https://scratch.mit.edu/ideas)



### Make sure students sign into their Scratch accounts

Have students sign into their own Scratch accounts at [scratch.mit.edu](https://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



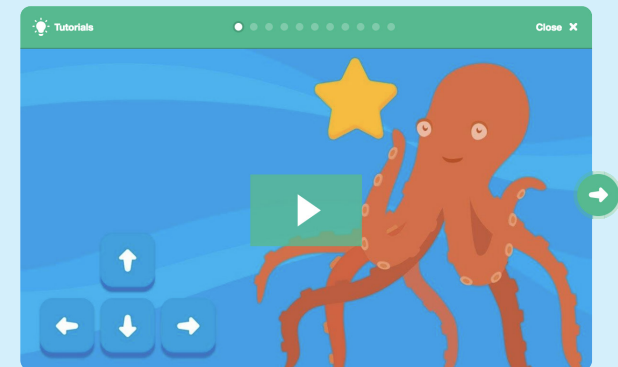
Begin by gathering the students to introduce the theme and spark ideas for projects.

### 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](https://scratch.mit.edu/chase)

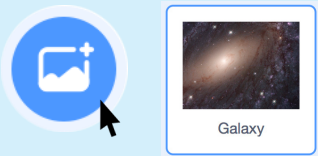


## Demonstrate the First Steps

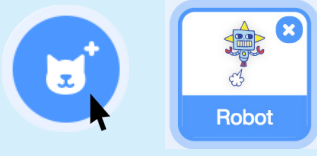


Demonstrate the first few steps of the tutorial so students can see how to get started.

**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

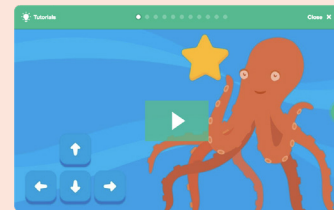
Ask students questions to get started

*Which backdrop would you like to choose for your game?*

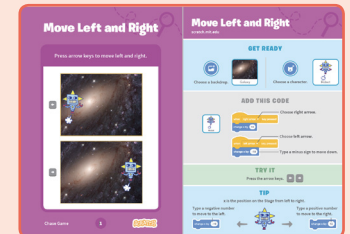
*Who do you want as the main character in your game? What will it chase?*

### Provide Resources

Offer options for getting started



Some students may want to follow the online tutorial: [scratch.mit.edu/chase](https://scratch.mit.edu/chase)



Others may want to explore using the printed cards: [scratch.mit.edu/ideas](https://scratch.mit.edu/ideas)

### Suggest Ideas for Starting

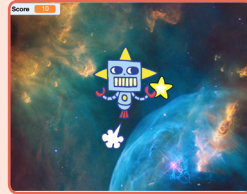
- Choose a backdrop
- Choose or draw a main character
- Make it move with arrow keys.
- Select an object to chase.



CREATE

### 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

# Share

Have students share their projects with their neighbors.

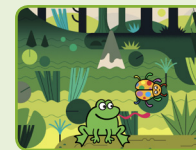
### Ask questions that encourage reflection:

*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.



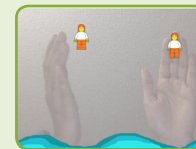
### Add Obstacles

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*

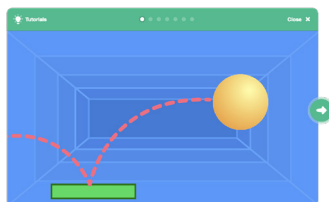
At the end of the session, gather together to share and reflect.

## Get Ready for the Lesson

Use this checklist to prepare for the lesson.

### Preview the Tutorial

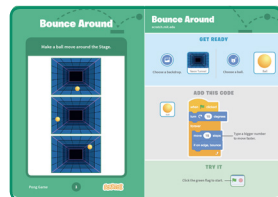
The *Pong Game* tutorial shows students how to create their own projects. Preview the tutorial before your lesson and try the first few steps: [scratch.mit.edu/pong](https://scratch.mit.edu/pong)



### 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](https://scratch.mit.edu/ideas)



### Make sure students sign into their Scratch accounts

Have students sign into their own Scratch accounts at [scratch.mit.edu](https://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.

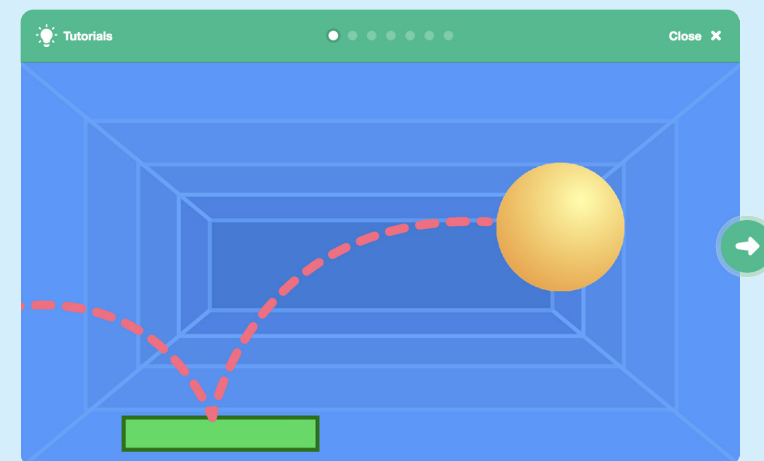
## Imagine



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](https://scratch.mit.edu/pong)

### Warm-up Activity: Bouncing Ideas

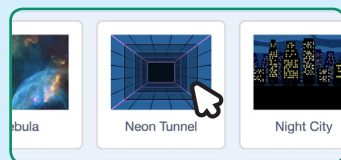
To get students thinking about a theme for their game, take turns calling out a theme, such as pizza pong or flower pong and brainstorming ideas for the type of images they could use to represent the theme.

## Demonstrate the First Steps



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

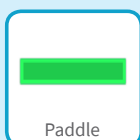
Go to the Scratch website. Click Create. Choose a new backdrop:



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

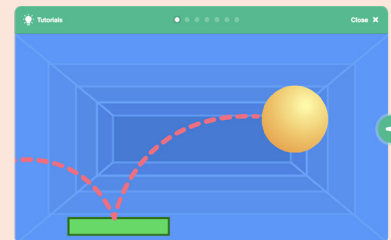
Ask students questions to get started

*What background do you want for your game?*

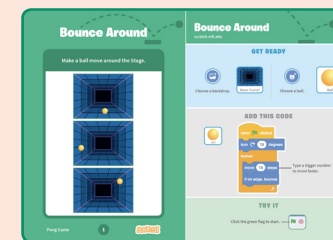
*What color or type of ball?*

### Provide Resources

Offer options for getting started



Some participants may want to follow the online tutorial:  
[scratch.mit.edu/pong](https://scratch.mit.edu/pong)



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

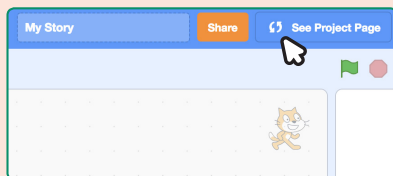
- Talk out what you're working on with someone
- Try out small bits of code at a time to figure out what's happening at each step
- Look closely at the blocks on the tutorial or activity cards to see if they are the same or different from the blocks you're using
- Look at the code for other pong games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

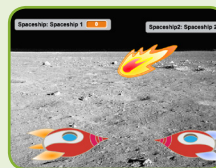
### Ask questions to encourage reflection:

*What did you notice about the games you tried?*

*What ideas might you add to your game?*

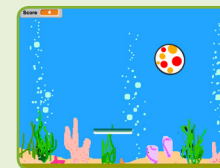
## What's Next?

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/](https://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.

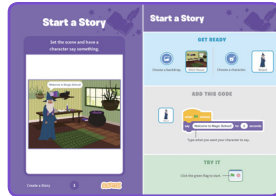
### Preview the Tutorial

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](https://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](https://scratch.mit.edu/ideas)



### Make sure students sign into their Scratch accounts

Have students sign into their own Scratch accounts at [scratch.mit.edu](https://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](https://scratch.mit.edu/story)

## Demonstrate the First Steps



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

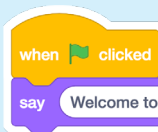
In Scratch, click Create.  
Choose a backdrop.



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



Code your character to say something.



Type what you want  
your character to say.

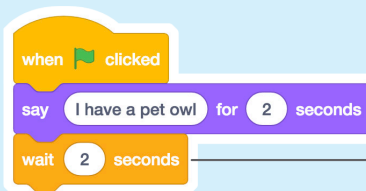
Click the green flag to start.



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?*

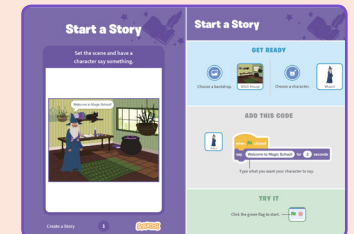
*What will  
happen first?*

### Provide Resources

Offer options for getting started



Some students may want to follow the online tutorial:  
[scratch.mit.edu/story](https://scratch.mit.edu/story)



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

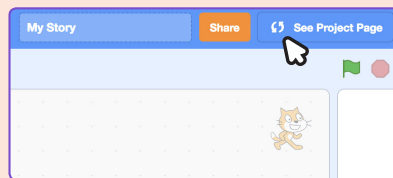
Scratch is designed to support creating by experimenting and tinkering. So, your students may want to start their stories without planning beforehand. As they create, one idea can spark another. Celebrate their sparks of creativity and the unexpected turns their stories may take.



### 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