

# Table of Contents

introduction	1.1
preCoder	1.2
coder	1.3
level 2	1.3.1
sequencing pixels	1.3.1.1
write some code	1.3.1.2
write read repeat	1.3.1.3
pixel bot online	1.3.1.4
code.org maze	1.3.1.5
dance dance js	1.3.1.6
pixel bot js	1.3.1.7
winter is coming - gather food	1.3.1.8
space ranger and magic words	1.3.1.9
coding arguments	1.3.1.10
level 3	1.3.2
sequencing pixels JS	1.3.2.1
write some code	1.3.2.2
write read repeat	1.3.2.3
pixel bot online	1.3.2.4
winter is coming - gather food	1.3.2.5
space ranger and magic words	1.3.2.6
coding arguments in JS	1.3.2.7
practicing arguments	1.3.2.8
fire ice and squirrels	1.3.2.9
developer	1.4
level 1	1.4.1
Lesson 1: I am a coder	1.4.1.1

---

<a href="#">Lesson 2: Getting Started on Scratch</a>	1.4.1.2
<a href="#">Lesson 3: Maze Scavenger Hunt</a>	1.4.1.3
<a href="#">Lesson 4: Dance Off</a>	1.4.1.4
<a href="#">Lesson 5: In the Loop</a>	1.4.1.5

---

# Welcome to getCoding



Coding is so much more than a language. It's a tool for creating video games, music, and art. A tool for changing the world.

We believe that all students should be equipped with the coding skills they need to be creators and active citizens—not just consumers—in the new digital world.

Our coding curriculum gives you the tools you need to empower your students to be digital developers, artists, and citizens. Our curriculum is broken up into 3 stages:

1. **preCoder** - An introduction to coding concepts for students who do not yet have the language skills to read and write. (grades K-2)
2. **coder** - A slowly paced deep dive into core coding concepts through puzzle-based lessons. (grades 3-8)
3. **developer** - An exploration of the many applications of coding concepts through project- and product-based lessons. (grades 3-8)

Ideally students would start with the preCoder stage and move to the coder and developer stages in 3rd grade. However, teachers should feel free to grab units from preCoder, coder, and developer as they see fit.

**Let's getCoding!**

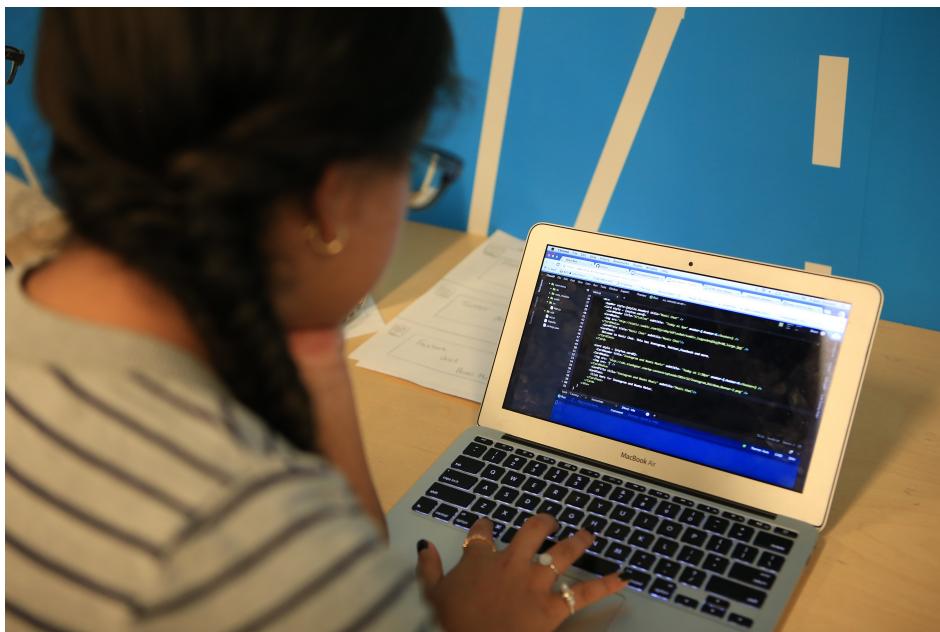
# preCoder

The preCoder series targets learners who cannot yet read and write. The series introduces and reinforces coding concepts through fun and engaging play-based curricula. Lessons include unplugged activities, worksheets, and online games.

# Curriculum

Coming Soon!

# Coder



The coder series features game-based curricula designed to introduce and reinforce coding concepts in a fun and engaging way for students. These units include unplugged activities where students can explore coding in a physical space, worksheets to help organize student thinking, and online games to practice coding skills.

## Curriculum

- Level 1 (Coming Soon!)
- [Level 2](#)
- [Level 3](#)

# Coder Level 2



Students are introduced to core coding concepts: sequences, calling functions, passing arguments, creating loops, and using conditionals. New concepts are introduced in a steady progression from an icon-based language to a block-based language and finally to typed JavaScript. This course includes lesson plans (unplugged and online), worksheets, and a custom-designed coding environment.

## Lessons

- [sequencing pixels](#)
- [write some code](#)
- [write read repeat](#)
- [pixel bot online](#)
- [code.org maze](#)
- [dance dance js](#)
- [pixel bot js](#)
- [winter is coming - gather food](#)
- [space ranger and magic words](#)
- [coding arguments](#)

## Download lesson plans

[Download](#)

## Workbook

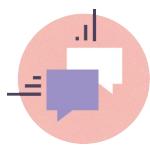
[Download](#)

## Answer key

[Download](#)



# Sequencing Pixels Unplugged



## OVERVIEW

In this lesson, students play with the order of commands in unplugged Pixel Bot exercises. The lesson explores a foundational concept in computer science—sequence.



## OBJECTIVES

---

- Know that computers run code in a sequence.
- Read, write, and execute code in a sequence.



## AGENDA

---

### Length: 45 minutes

Demonstrate how coders shape our world. (10 minutes) Present an array of examples of where we find code in the modern world. Consider autonomous cars, music, online communities. Watch "A Day in the Life of a Software Engineer" (<http://tinyurl.com/q966xd5>).

Unearth students' ideas about computer sequencing. (15 minutes) Using [Lesson 1 | Warm-up Worksheet](#): Students develop ideas about how the arrow and paint elements work in Pixel Bot. Students step through and enact the two programs in the worksheet. Gather students' ideas about computer sequencing and show how a slight change in sequence alters the outcome.

Demonstrate how to read a program like a computer. (10 minutes) Draw a Pixel Bot program and grid on the whiteboard (3 or 4 lines of code, 3 x 3 grid). Work with students to step through and enact the program. Show the usefulness of numbering lines of code. Repeat a few times with new programs.

Support students' practice of reading programs. (10 minutes)  
Using Worksheet 1 | [Page 1](#) & [Page 2](#): Students step through and enact the programs by tracing the route of the pixel bot and shading in any squares that it paints. (10 minutes)



## CONTENT KNOWLEDGE

---

- Sequence - The idea that statements must be performed in the order they are written.



## MATERIALS

1. [Lesson 1 | Warm-up Worksheet](#)
2. [Worksheet 1 | Page 1 & Page 2](#)
3. Small pixel bot cutout for each student
4. Magnetic pixel bot
5. Scratch paper grids
6. Pencils
7. Whiteboard
8. Queued up video: <http://tinyurl.com/q966xd5>



## WELCOME TO CODING



Length: 10 minutes

Teacher Actions	Student Actions
<p><b>1</b> Lead a discussion:</p> <div style="background-color: #f0f0f0; padding: 10px; margin-bottom: 10px;"> <p>What does it mean to be a coder?</p> </div> <div style="background-color: #f0f0f0; padding: 10px; margin-bottom: 10px;"> <p>Where do we use code?</p> </div> <p>Chart responses on the whiteboard.</p>	<p><b>1</b> Coders:</p> <div style="background-color: #f0f0f0; padding: 10px; margin-bottom: 10px;"> <p>Work on computers Hack things Create video games Make websites Work with data</p> </div> <p>Code runs in:</p> <div style="background-color: #f0f0f0; padding: 10px; margin-bottom: 10px;"> <p>Phones Computers Traffic lights Spaceships Game consoles Movies and tv shows</p> </div>
<p><b>2</b> Offer other examples:</p> <div style="background-color: #f0f0f0; padding: 10px; margin-bottom: 10px;"> <p>Autonomous cars Streetlights Music Flight simulators And on and on...</p> </div>	
<p><b>3</b> Watch video: A day in the life of a software engineer:  <a href="http://tinyurl.com/q966xd5">http://tinyurl.com/q966xd5</a></p>	



## PREDICT PIXEL BOT ICONS



**Length:** 15 minutes

Teacher Actions	Student Actions
<p><b>1</b> Distribute [Lesson 1   Warm-up Worksheet][warm-up].</p>	
<p><b>2</b> Introduce reading code with care:</p> <p>Imagine being a computer when you read code.</p> <p>As computers, we read carefully. We pay attention to every detail. Every line of code .</p>	
<p><b>3</b> Have students interpret the coding elements in the warm-up:</p> <p>What do you think</p>	<p><b>3</b> Possible Responses:</p> <p>The bot rotates i</p>

## sequencing pixels

<p>these elements do?</p>	<p>n place. The bot jumps. The bot moves until it hits the edge. The bot shoots lightning.</p>	<p>Correct Responses:</p> <p>The bot moves one square at a time. The bot moves up, right, left, and down. The bot paints.</p>
<p>4 Individual Work: Have students fill out the worksheet.</p>	<p>4 Trace the movement of the Bot. Paint with the Bot.</p>	
<p>5 While students code, draw the worksheet's programs and grids on the whiteboard.</p>		
<p>6 Reflect on sequence:</p> <p>How did you arrive at your answer?</p> <p>What is the difference between the two programs?</p> <p>How does the order</p>	<p>6 I arrived at my answer by:</p> <p>Reading one element at a time. Moving the bot after each element. Following the correct sequence.</p> <p>The two programs:</p> <p>Have elements in</p>	

## sequencing pixels

<p>r of the icons ma tter?</p>	<p>different orders.</p> <p><b>Order matters because:</b></p> <p>The bot paints a different square.</p>
<p><b>7</b> Solve the two warm-up problems together on the whiteboard.</p>	<p><b>7</b> Call out lines of code and bot actions.</p>
<p><b>8</b> Work as a class to define each element.</p>	<p><b>8</b> Work as a class to figure out what each element makes the bot do.</p>



## DEMONSTRATE HOW TO READ AND STEP THROUGH PROGRAMS



Length: 10 minutes

Teacher Actions	Student Actions
<p>1 Draw a blank 3x3 grid on the whiteboard.</p>	
<p>2 Write a short (3 or 4 line) program on the whiteboard.</p>	
<p>3 Explain sequence:</p> <p>When a computer executes code, it steps through the code in the correct order. This is called sequencing.</p>	
<p>4 Read the first line of code together:</p> <p>What is the first</p>	<p>4 Students call out the first line of code.</p>

## sequencing pixels

line of code?  Number the first line of code.	
5 Move the Pixel Bot after reading each line. Trace its path or shade in a square.	5 Students call out where Pixel Bot should move.
6 Continue reading and stepping one line at a time.	6 Students continue helping to read the code.
7 Read and step through three new examples with the class. Design problems on the fly, making them interesting and complex enough.	



## READ PIXEL BOT ICONS

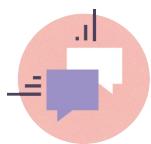


**Length:** 5 minutes

Teacher Actions	Student Actions
<p><b>1</b> Distribute Worksheet 1: [Page 1][worksheet1-1] &amp; [Page 2] [worksheet1-2].</p>	
<p><b>2</b> Leave the worked example from the previous activity on the whiteboard.</p>	
<p><b>3</b> Individual Work: Ask students to individually fill out the worksheet.</p>	<p><b>3</b> Students read the code, trace the pathway of the pixel bot, and paint the correct blocks on the worksheet.</p>



# Write Some Code Unplugged



## OVERVIEW

Students learn the different components of reading and writing code by exploring the roles (writer, reader, navigator, and stepper) of Coders & Bots.



## OBJECTIVES

---

- Students will be able to read basic code.
- Students will be able to write basic code.
- Together as a class, students will be able to enact the Coders & Bots roles of Writer, Navigator, Reader, and Stepper to write and read code.



## AGENDA

---

Length: 45 mintues

1. Pixel Bots: Practice Reading Code
2. Pixel Bots: Write Code
3. Pixel Bots: Write Code Together



## VOCAB

---

- Computer - A device that can be instructed to do something.
- Program - A list of statements that a computer can perform.



## MATERIALS

---

write some code

---

1. [Lesson 2 | Warm-up Worksheet](#)
2. [Lesson 2 | Worksheet 1](#)
3. Small pixel bot cutout for each student
4. Magnetic pixel bot
5. Scratch paper grids
6. Pencils
7. Whiteboard



## PIXEL BOTS: PRACTICE READING CODE



Length: 15 minutes

Students practice reading basic block code sequences.

Prep: Draw on the whiteboard the grid and the lines of code from the example problem on the front page of [Lesson 2](#) |

[Warm-up Worksheet](#)

Teacher Actions	Student Actions
<p>1 Remind students of the value of revisiting ideas explored in previous lessons. Revisiting past ideas helps to see them in a new light and makes sure they are not forgotten.</p>	
<p>2 Step through the example problem as a whole class, talking through the process of reading code.</p>	
<p>3</p>	<p>3</p>

## write some code

Hand out [Lesson 2   Warm-up Worksheet][warm-up].	Students place their pixel bot at the starting square. Students read the code and move their pixel bot. Students do this a few times to get into a rhythm.
4 Individual Work: Ask students to place the movable pixel bot at the start block of the example problem just demonstrated on the board. Students should work individually, reading the code and moving their pixel bot along. Ask the students to get into a rhythm of reading and stepping.	
5 Individual Work: Ask students to flip the page and perform the same exercise with the new problem. Ask students to shade in the appropriate squares and trace the pathway of the pixel bot.	5 Students turn the page and attempt to solve the problem.

6

On the board, draw the empty grid and the code from the problem students have been working on. Read the code and step the pixel bot (tracing its pathway and shading in squares), narrating your thought process.



## PIXEL BOTS: WRITE CODE



Length: 10 minutes

Students write code to produce a simple pixel bot image.

Prep: Hand out [Lesson 2 | Worksheet 1](#).

Teacher Actions	Student Actions
<p>1 Talk to students about how they have already learned to read code that other people have written. Now they are going to explore writing code.</p>	
<p>2 Individual Work: Have students place their pixel bot at the start square. Ask students to write code that produces the provided pixel bot image. The students should think through their plan for their code, write their code, and move their pixel bot along.</p>	<p>2 Students place their pixel bot at the starting square, write code that creates the picture, and enact the pixel bot actions each step of the way.</p>

3

As a whole class, solicit ideas from multiple students about how they designed their code to solve the problem. Hold off on presenting the correct answer until the next activity.

3

Students share their code, especially if they have solved the problem a different way.



## PIXEL BOTS: WRITE CODE TOGETHER



Length: 20 minutes

Teacher shows student how to write code, emphasizing roles of the Writer, Navigator, Reader, and Stepper (see Coders and Bots Protocol). The students then enact these roles together as a whole class solving a new problem.

Prep: Draw the problem from [Lesson 2 | Worksheet 1](#) on the whiteboard.

Teacher Actions	Student Actions
<p>1 Code the solution to [Lesson 2   Worksheet 1] [worksheet1] on the whiteboard. Voice your code writing process along the way: write and number a new line of code and only then move the pixel bot on the whiteboard. Use this as an opportunity to discuss how there are different ways to solve the same problem. After the code is written, introduce the idea</p>	

that the computer reads code in sequence.

**2** After coding the solution, explain that you are switching gears into Bot mode. Read each line and step the pixel bot, checking to see if you coded the correct solution. Add a new problem on the board: a checkerboard pattern. Divide the class in half and follow the process spelled out in the whole class section of the Coders and Bots Protocol. Students start as Coders. Assign half of the students to be writers and half to be navigators; one person from each team walks up to collaboratively add and enact a line of code. The two students then tag in a member of their team to walk up and write the next line of code. Allow the class to code a wrong solution.

**2** Students take turns walking up to the board to write and navigate code.

<p>3 The whole class then switches roles to become Bots. Use this opportunity to emphasize the definition of a computer (see vocabulary section above). Following the same procedure as above, ask students to come up to the board to take turns reading and stepping through one line of code at a time before tagging in a new classmate.</p>	<p>3 Students take turns walking up to the board to read and step through the solution.</p>
<p>4 Have students switch back and forth between Coders and Bots until they code the solution on the checkerboard.</p>	<p>4 Students switch back and forth between Coders and Bots to complete the puzzle.</p>
<p>5 Summarize any conceptual difficulties. Introduce and define a Program (see vocabulary section above), and explain how it connects to the code the students just wrote.</p>	

<p>6 Ask students to write down on a piece of paper what each of the four roles does, focusing on one role at a time. After each role definition, ask students to share their definition with the whole class. Pool the students' ideas into overall definitions of the roles.</p>	<p>6 Students provide descriptions of what each role entails.</p>



**Write.Read.Repeat.**  
Unplugged



## OVERVIEW

Students continue to solve pixel bot problems by playing Coders & Bots to read and write code.



## OBJECTIVES

---

- Students will perform the Coder & Bots roles of Writer, Navigator, Reader, and Stepper
- Students will collaborate in small groups to write and read code
- Students will become more proficient at writing code



## AGENDA

---

Length: 45 minutes

1. Pixel Bots: Write Code Warm-up
2. Pixel Bots: Coders and Bots
3. Pixel Bots: Write Code Exit Ticket



## VOCAB

---

- Programming Element - A command the computer understands.
- Action - An observable event that the code produces
- Error - The program fails to run because the computer cannot execute the code



## MATERIALS

---

1. Lesson 3 | Warm-up Worksheet
2. Lesson 3 | Worksheet 1
3. Lesson 3 | Worksheet 2
4. Lesson 3 | Worksheet 3
5. Lesson 3 | Worksheet 4
6. Lesson 3 | Exit Ticket Worksheet
7. Scratch paper grids
8. Small pixel bot cutout for each student
9. Magnetic pixel bot
10. Scratch paper grids
11. Pencils
12. Whiteboard



## CLASSROOM SETUP

---

Pods of four.



# PIXEL BOTS: WRITE CODE WARM-UP



**Length:** 15 minutes

Students write code for a simple pixel bot image.

**Prep:** Hand out Lesson 3 | Warm-up.

Teacher Actions	Student Actions
<p><b>1</b> Individual Work: Ask students to solve the problem on [Lesson 3   Warm-up Worksheet][warm-up].</p>	<p><b>1</b> Students individually solve the problem on the worksheet.</p>
<p><b>2</b> As students attempt the problem, draw the problem on the whiteboard.</p>	
<p><b>3</b> Code the solution to the problem as a whole class. Call on students at random to provide each next line of code. Each student should walk up to the board and draw the</p>	<p><b>3</b> If called on, students walk up to the board to write in (or read) the next line of code.</p>

next symbol. With each new symbol, you should play the role of Navigator, moving the turtle. Call out the roles of Writer and Navigator to make them explicit. On occasion, pick a Reader and Stepper to walk up to the board to test the code starting from line one.



## PIXEL BOTS: CODERS AND BOTS



**Length:** 30 minutes

Students work in groups to write and read code to produce two pixel bot images.

**Prep:** Consider having paper bags for each group. Each paper bags contains the four roles (Writer, Navigator, Reader, Stepper).

Teacher Actions	Student Actions
<p>1 Teacher breaks students into groups of four and randomly assigns roles for the Programming Team and the Computer Team. (Consider handing each group a paper bag with the four roles inside – students reach into the bag and grab a role.) In this first round, groups either get [Lesson 3   Worksheet 1] [worksheet1] or [Lesson 3   Worksheet 2] [worksheet2] (alternate between groups). Bots should always help the Coders during</p>	<p>1 Students enact their Coders and Bots roles. They write code to create the pixel bot image.</p>

the code writing phase of the activity.

2

Follow the Coders and Bots Protocol by having the Bot Team switch to a new group when it comes time to check the code. Make sure that the Coders fold back their paper to hide the provided pixel bot image before the Bots arrive (ensuring that the Bots are not biased in their reading). The Bots should test the code on an empty scratch paper grid. Ask students, "What do you think the computer does when it cannot understand the code or the code forces the turtle to break the rules (go outside the grid)?" Answer: An error! If the Bots notice this happening, they should report the error to the programming team and tell them what line the error happened on.

2

The Bots switch to a new group to assess code. If the Bots find an error, report it to the Coders and tell them what happened and the line number the error happened on.

<p>3 After one round, pass out [Lesson 3   Worksheet 3] [worksheet3] or [Lesson 3   Worksheet 4] [worksheet4] to the groups (alternate again) and repeat the Coders and Bots Protocol.</p>	<p>3 Students repeat the above process with new coding challenges.</p>	
<p>4 In whole class mode, ask students to share out any disagreements they had in the write and read process. Use this as an occasion to firm up any misconceptions. Also use this time to introduce and define Elements and Actions (see vocabulary section above).</p>	<p>4 Students summarize and describe the disagreements they could not resolve.</p>	



## PIXEL BOTS: WRITE CODE EXIT TICKET



**Length:** 5 minutes

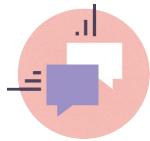
Time permitting, students individually fill out an exit ticket to check for understanding.

**Prep:** Hand out [Lesson 3 | Exit Ticket Worksheet](#)

Teacher Actions	Student Actions
<p>1 Ask students to individually work on completing the [Lesson 3   Exit Ticket Worksheet] [wrap-up].</p>	<p>1 Students work individually on the [Lesson 3   Exit Ticket Worksheet] [wrap-up].</p>



Pixel Bots Online  
Plugged



## OVERVIEW

Students review writing code offline. Then, students demonstrate their learning by writing programs to complete online pixel bot challenges.



## OBJECTIVES

---

- Students will write programs using an icon language.
- Students will continue to develop proficiency in writing and reading code.



## AGENDA

---

**Length: 45 mintues**

1. Pixel Bots: Warm-up
2. Introduce Pixel Bots Online
3. Pixel Bots Online Practice
4. Pixel Bots: Exit Ticket



## VOCAB

---

- Program - A list of statements that a computer can perform.



## MATERIALS

---

1. Lesson 4 | Worksheet 1

2. [Lesson 4 | Exit Ticket Worksheet](#)
3. Scratch paper grids
4. Small pixel bot cutout for each student
5. Magnetic pixel bot
6. Scratch paper grids
7. Pencils
8. Whiteboard



## PIXEL BOTS: WARM UP



Length: 10 minutes

Create a medium difficulty pixel bot image for the students on the whiteboard. Students solve the problem individually and then check the work of a peer.

Prep: Create a medium difficulty pixel bot image on the whiteboard.

Teacher Actions	Student Actions
<p>1 Individual Work: Ask students to get a blank piece of paper, write line numbers, and write the code that creates the pixel bot image on the board.</p>	<p>1 Students work individually on coding a solution to the challenge.</p>
<p>2 Students discuss the coding solution with a peer.</p>	<p>2 When they finish, students check their work with a partner after both have finished.</p>



# INTRODUCE PIXEL BOT ONLINE



Length: 10 minutes

Show students how to solve a problem on pixel bot online. Also revisit **program** as a vocabulary word.

Prep: Browse to [pixelbots.io](https://pixelbots.io) and project onto the wall.

Teacher Actions	Student Actions
<p>1 Now that students have practice creating code sequences on paper, they are ready to start writing programs.</p>	<p>1 Add student actions here and match numbers to teacher actions</p>
<p>2 Revisit the idea that a program is a sequence that a computer is able to understand.</p>	
<p>3 On a projector, show students <a href="https://pixelbots.io">pixelbots.io</a></p>	
<p>4</p>	

Use the problem you created in the warm-up to demonstrate how to solve the problem in the online interface. Show students a variety of features in the pixel bot interface:

- How to add code
- How to delete code
- How to run program
- How to reset after run
- How to insert code

5

Ask students, "What do you think we should do if we get stuck on a problem?"

5

Answer: Step through code starting at the beginning like you do as a Bot.



## PIXEL BOT ONLINE PRACTICE



Length: 20 minutes

Students are given a set of pixel bot images to reproduce on the computer. Partners star the ones that are completed. Remind students that the images with lots of shaded squares will be difficult, but they should remember to read code from the beginning when things go wrong.

Prep: Distribute the [Lesson 4 | Worksheet 1](#) and write [pixelbots.io](#) up on the board.

Teacher Actions	Student Actions
<p>1 Explain the exercise:</p> <ul style="list-style-type: none"><li>• The goal is to recreate each of the images from the worksheet by creating a program on the website.</li><li>• Tell students that they will act as testers for the person sitting next to them:<ul style="list-style-type: none"><li>◦ When they finish writing a program, students have their</li></ul></li></ul>	<p>1 Students are faced away from their computers toward the teacher.</p>

partner check to make sure the images match up. If they are a match, the tester puts a checkmark next to the image.

- o The programmer should then explain how their code works to the tester. If the tester is satisfied with the explanation, the tester checks the explain box.
- o The programmer can now continue on to the next challenge.

2

Students work on completing the challenges.

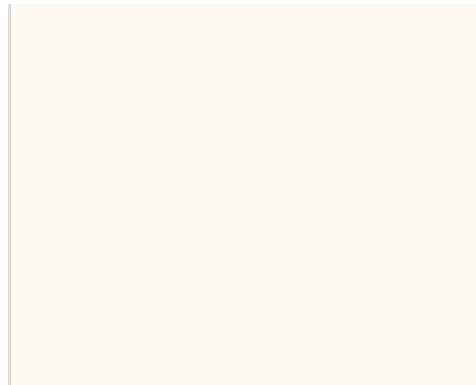
2

Students get on their computers and go to [pixelbots.io](https://pixelbots.io). Students work individually on

	<p>recreating each of the images with code.</p> <ul style="list-style-type: none"><li>When they finish a puzzle, students have their partner (student sitting next to them) check to make sure the images match up. If they are a match, the checker puts a checkmark next to the image and the programmer can continue on to the next challenge.</li></ul>
3	When students get stuck, ask them to imagine being on the Coder team from the group activities. They should try to play the roles of the writer and navigator. Then, ask students to imagine being a Bot to understand how the computer is reading the code.
3	Students raise their hands to provide answers.

4

Discuss: Which was the hardest coding challenge? Why? Was there more than one way to solve the problem?





## PIXEL BOTS: EXIT TICKET



**Length:** 5 minutes

Students complete Exit Ticket.

**Prep:** Distribute [Lesson 4 | Exit Ticket Worksheet](#).

Teacher Actions	Student Actions
<p><b>1</b> Individual Work: Tell students they are going to work individually on coding the answer on the worksheet.</p>	<p><b>1</b> Students write their answers on the worksheet.</p>



Code.org Maze  
Plugged



## OVERVIEW

Students will build on their understanding of algorithms learned in Pixelbots to solve problems with different commands in the Code.org environment.



## OBJECTIVES

---

- Students will decompose problems with new elements
- Students will learn to use a block-based language



## AGENDA

---

**Length:** 45 minutes

1. Solving problems with new code elements
2. Introduce code.org maze
3. Code.org Maze: Deliberate practice



## VOCAB

---

**Programming Language** - A set of programming elements used to communicate to a computer.



## MATERIALS

---

1. Scratch paper grids
2. Small pixel bot cutout for each student

3. Magnetic pixel bot
4. Scratch paper grids
5. Pencils
6. Whiteboard



## CODE ELEMENT CHANGE-UP



Length: 15 minutes

Students are given different code elements to solve similar pixel bots problems.

Prep: Draw a medium difficulty coding challenge on the board.

Teacher Actions	Student Actions
<p>1 Draw the normal icon elements students are accustomed to and ask students to use them to solve the puzzle.</p>	<p>1 Students use the elements to develop a solution.</p>
<p>2 Discuss student solutions.</p>	<p>2 Students raise their hands to offer solutions.</p>
<p>3 Introduce students to a different set of elements (degree turns – see the arrows below) by drawing them on the board. What do students think they mean? How much should they turn</p>	<p>3 Students raise their hands to discuss the new elements.</p>

right? How much should they turn left? How far should they move forward?

- - move one block forward
- - turn right 90 degrees
- - turn left 90 degrees

4

Students solve the puzzle using the new code elements.

4

Students solve the problem using the new elements.

5

Discuss: What was the difference between the elements? How do the elements you have affect your solution?

5

Students raise their hand to discuss the difference when using new elements.



# INTRO TO CODE.ORG MAZE



**Length:** 5 minutes

Introduce students to code.org by projecting the teacher screen and solving a maze puzzle as a class.

**Prep:**

1. Set up projector
2. Navigate to  
<https://studio.code.org/s/course2/stage/3/puzzle/1>.

Teacher Actions	Student Actions
<p>1 Point out that code.org uses a different programming language.</p> <ol style="list-style-type: none"><li>1. Ask students to look at the available code elements (on the left side of the editor) and talk about what is different from pixelbots.io</li><li>2. A programming language is how a person can communicate with a computer.</li></ol>	<p>1 Students offer answers about the differences between the elements in pixelbots.io and code.org</p>

<p>There are many programming languages.</p> <p>3. Pixel bots used an icon language. Code.org uses a block based language.</p>	
<p>2 students the projected screen and show them how to use the code.org interface:</p> <ul style="list-style-type: none"><li>• How to drag and drop blocks into the code editor</li><li>• How to run the code</li><li>• How to reset after a run</li></ul>	<p>2 Students are off the computer and facing the teacher</p>
<p>3 As a class solve the first puzzle.</p>	<p>3 Students raise their hands to offer answers about what blocks to add to the code.</p>



# CODE.ORG MAZE: WRITE CODE PRACTICE



Length: 30 minutes

Students do the rest of the Code.org maze puzzles on their own.

Teacher Actions	Student Actions
<p>1 Explain the exercise and facilitate students navigating their browsers to the correct place.</p> <ol style="list-style-type: none"><li>1. Students navigate to studio.code.org</li><li>2. Click on course 2</li><li>3. Click on the first Stage 3: Maze: Sequence</li></ol>	<p>1 Students open their web browsers and navigate to the challenges</p> <ol style="list-style-type: none"><li>1. Students navigate to studio.code.org</li><li>2. Click on course 2</li><li>3. Click on the first Stage 3: Maze: Sequence</li></ol>
<p>2 Tell students to work through as many of the exercises as they can.</p>	<p>2 Students work on completing the mazes.</p>
<p>3 When students get stuck, use the Writing, Reading, and Debugging</p>	

protocols to support student learning. Ask students to imagine being on the Coder team from the group activities. They should try to play the roles of the writer and navigator. Then, ask students to imagine being a Bot to understand how the computer is reading the code.



# Dance Dance JS Unplugged



## OVERVIEW

Students will learn the JavaScript syntax for a function call and apply that knowledge and their knowledge of sequencing to write real JavaScript programs.



## OBJECTIVES

---

- Students will learn the syntax for a function call.
- Students will continue to develop proficiency in writing and reading code.
- Students will develop pairs programming proficiency.
- Students will work in teams to solve problems through the use of the Coders and Bots Protocol.



## AGENDA

---

**Length: 45 minutes**

1. Offline Pixel Bot Warm-up (5 minutes)
2. Pixel Bot JS Intro to Function Calls (15 minutes)
3. Dance Dance JS (25 minutes)



## VOCAB

---

- **Function call** - A programming element that tells the computer to do something. In the beginning, most function calls will cause the computer to perform an action.
- **Syntax** - The way a programming element should be written.
- **Programming language** - A set of elements used to

communicate with a computer.



## MATERIALS

---

1. [Lesson 6 | Warm-up Worksheet](#)
2. [Lesson 6 | Worksheet 1](#)
3. [Lesson 6 | Worksheet 2](#)
4. Scratch paper grids
5. Small turtle cutout for each student
6. Magnetic turtle
7. Scratch paper grids
8. Pencils
9. Whiteboard



## OFFLINE PIXEL BOT WARM-UP



**Length:** 5 minutes

Students warm up their code writing skills by performing a medium difficulty pixel bot exercise.

**Prep:** Hand out [Lesson 6 | Warm-up Worksheet](#).

Teacher Actions	Student Actions
<p>1 Ask students to solve the problem on the [Lesson 6   Warm-up Worksheet][warm-up].</p>	<p>1 Students complete the warm-up.</p>



## INTRODUCE SYNTAX



Length: 15 minutes

Students perform the same exercise as in the warm-up, but the programming elements are replaced with real JavaScript syntax. Teacher emphasizes the importance of getting the syntax right.

Prep: Hand out [Lesson 6 | Worksheet 1](#).

Teacher Actions	Student Actions
<p>1 Point out to students that the programming elements changed but that they may still be able to solve the problem.</p>	
<p>2 Individual Work: Ask students to attempt the problem in [Lesson 6   Worksheet 1] [worksheet1].</p>	<p>2 Students attempt to complete [Lesson 6   Worksheet 1] [worksheet1].</p>
<p>3 Explain that these new programming elements are part of JavaScript.</p>	

4 Explain that these particular programming elements are all function calls and that we know they are function calls because they have an open and closed parenthesis after the name.

5 Walk through the solution to the worksheet on the board. Highlight the importance of including the parenthesis.

6 Ask students: What would happen if I did not include the parenthesis after the `up`? Answer: nothing, the computer would just skip that line.



## DANCE DANCE



Length: 25 minutes

Teacher performs dance based on dance program that only she/he can see. Then students try to figure out the code for the dance by playing Coders and Bots. If there is extra time, teams can write their own programs for dances.

Prep: Hand out [Lesson 6 | Worksheet 2](#).

Teacher Actions	Student Actions
<p>1 Explain how Dance Bot JS works (see activity directions below).</p>	
<p>2 Explain that you will be performing a dance and that the students will need to figure out what code programmed your dance. Luckily students will have their own dance bots to test the code on, because they will be playing Coders and Bots. All the roles will be the same for Coders and Bots as it was for the Pixel Bot</p>	

<p>exercise except that the stepper will be dancing.</p>	
<p>3 Explain that every time the Bots switch tables, she/he will perform the dance again.</p>	
<p>4 Ask students to pick their roles.</p>	
<p>5 Perform your dance.</p>	
<p>6 Let Coder and Bots progress, trying to figure out the code that guided the dance, until teams have produced the correct program.</p>	<p>6 Students start playing Coders and Bots to come up with the code for the dance. They can write the code on a blank [Lesson 6   Worksheet 2] [worksheet2].</p>
<p>7 Individual Work: Tell students that they will now be creating their own dance program (5 minutes). Ask each</p>	<p>7 Students individually write the code for their own dance using a blank [Lesson 6   Worksheet 2] [worksheet2].</p>

<p>student to write his/her own dance program.</p>	
<p>8 Tell the readers and writers at each table to perform their partner's (steppers &amp; navigators) code.</p>	<p>8 Readers and writers dance their partner's code.</p>
<p>9 Tell the steppers and navigators at each table to perform their partner's code.</p>	<p>9 Steppers and navigators dance their partner's code.</p>



## DANCE BOT JS

Dance bot is a game in which a dance bot is programmed to perform a dance. The movement of the dance bot is very similar to players in dance dance revolution. Dance bot is a fun way for students to continue to explore the importance of sequencing and shows students how code and art can intersect.



### ELEMENTS

---

- `up()` - move either foot up and return it
- `left()` - move left foot left and return it
- `right()` - move right foot right and return it
- `down()` - move either foot down and return it
- `spinLeft()` - spin a quarter turn left
- `spinRight()` - spin a quarter turn right
- `wait()` - don't do anything for a beat



### RULES

---

All commands should be executed on a beat. It is the job of the reader of the program to keep the beat.

The dance bot operates in a 9x9 grid (real or imaginary). It starts each move from the center of the grid. This means the dance bot should never move outside of the 9x9 grid - the dance is essentially performed in place.



### EXAMPLE DANCE

---

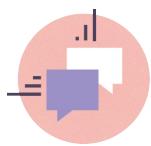
## dance dance js

---

```
up()
up()
left()
right()
up()
up()
left()
right()
spinLeft()
up()
up()
right()
left()
up()
up()
right()
left()
spinLeft()
back()
back()
left()
wait()
back()
back()
right()
wait()
spinLeft()
spinLeft()
left()
right()
```



Pixel bot js  
Plugged



## OVERVIEW

Students learn about how to type the special symbols required for JS and use Javascript on pixelbot.io to solve problems.



## OBJECTIVES

---

1. Students will learn to identify syntax errors
2. Students will learn how to input special characters on the computer
3. Students will continue to develop proficiency in writing and reading code.



## AGENDA

---

**Length: 45 minutes**

1. Unplugged Warm-up (5 minutes)
2. Start Coding
3. Pair Idea Exchange
4. Continue Coding



## VOCAB

---

**syntax error** - An error from typing the JS language incorrectly



## MATERIALS

---

1. [Lesson 7 | Worksheet 1](#)
2. [Lesson 7 | Exit Ticket Worksheet](#)
3. Scratch paper grids
4. Small turtle cutout for each student
5. Magnetic turtle
6. Scratch paper grids
7. Pencils
8. Whiteboard



## WARM-UP



**Length: 5 minutes**

Students revisit JS in an unplugged challenge.

Prep: Draw your own pixelbot challenge on the board.

Teacher Actions	Student Actions
<p><b>1</b> Individual Work: Tell students to solve the pixelbot challenge in JavaScript.</p>	<p><b>1</b> Students write down their solutions on scratch paper.</p>
<p><b>2</b> Discuss solutions. Ask students, what would happen if I only added the open parenthesis after the <code>right</code> like <code>right( ?</code></p> <p>Answer: The computer would throw a syntax error and the program would not run.</p>	<p><b>2</b> Students raise their hands to provide answers.</p>



## START CODING



Length: 35 minutes

Students learn how to input special characters on the keyboard and are given a series of images to recreate using pixelbot.

Prep: Distribute [Lesson 7 | Worksheet 1](#)

Teacher Actions	Student Actions
<p>1 Instructions</p> <ol style="list-style-type: none"><li>1. The goal is to recreate the images on pixelbot.io</li><li>2. Ask students, "What are the special symbols you need to call a function?" Answer: parentheses</li><li>3. Show students where the parentheses are located on the computer and remind them that you need to hold down shift to use them.</li><li>4. How many elements (function calls) should be on</li></ol>	<p>1 Students turn away from computers and face the teacher.</p>

each line? How do you go to the new line?

Answer: There should be 1 element per line.

5. Show students where the return key is on the keyboard
6. Show student an example of a syntax error on pixelbot.io. Point out the red 'x' that appears in the editor when you add the error.

3

When students get stuck, we suggest using the Read, Write, and Debug protocols to support students. Ask students to imagine being on the Coder team from the group activities. They should try to play the roles of the writer and navigator. Then, ask students to imagine being a Bot to understand how the computer reads the code.

pixel bot.js

---



## UNPLUGGED EXIT TICKET



**Length: 5 minutes**

Students finish a JS worksheet to demonstrate learning

**Prep:** Distribute [Lesson 7 | Exit Ticket Worksheet](#)

Teacher Actions	Student Actions
<p><b>1</b> Individual Work: Ask students to solve the exit ticket.</p>	<p><b>1</b> Students complete the exit ticket to the best of their abilities.</p>



**Gather Food - Winter is Cor  
plugged**



## OVERVIEW

Students write JavaScript on getcoding.io to solve increasingly difficult challenges involving moving a squirrel to gather nuts.



## OBJECTIVES

---

1. Students write basic JavaScript to solve simple navigation problems.



## AGENDA

---

**Length:** 45 minutes

1. Warm-up
2. Help the Squirrel Gather Acorns
3. Pair Idea Exchange
4. Resume Helping the Squirrel Gather Acorns
5. Unplugged Exit Ticket



## VOCAB

---

**Code Editor** - The place where coders assemble their program.



## MATERIALS

---

1. [Lesson 8 | Warm-up Worksheet](#)
2. [Lesson 8 | Exit Ticket](#)

3. Laptops/Computers
4. Scratch paper grids
5. Small turtle cutout for each student
6. Magnetic turtle
7. Scratch paper grids
8. Pencils
9. Whiteboard



## WARM-UP

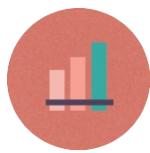


Length: 10 minutes

Students practice writing basic javascript to create a simple Pixel Bot drawing.

Prep: Hand out [Lesson 8 | Warm-up Worksheet](#)

Teacher Actions	Student Actions
<p><b>1</b> Individual Work: Ask students to write the code to produce the Pixel Bot image in the [Lesson 8   Warm-up Worksheet] [warm-up]. Consider reminding students of the proper JavaScript syntax (see Elements on the worksheet).</p>	<p><b>1</b> Students individually fill out the Warm-up Worksheet.</p>
<p><b>2</b> Draw the Pixel Bot image on the whiteboard and code the solution with the students, randomly calling on one student at a time to provide each next line of code. (Note the problem can be solved in different ways. Students should follow the class' ongoing code which may differ from their own solution).</p>	<p><b>2</b> If called on, students call out the next line of code.</p>



## HELP THE SQUIRREL GATHER ACORNS



**Length: 15 minutes**

Students write code in JavaScript on the getcoding.io platform to help the squirrel gather acorns. Students are practicing simple sequences.

**Prep:** Have the getcoding.io platform open on your browser and projected on the wall. Students should also have their own computers.

Teacher Actions	Student Actions
<p>1 Walk students through the getcoding.io platform. Show students how to:</p> <ul style="list-style-type: none"><li>• open activities</li><li>• use the Code Editor (where to type)</li><li>• see elements</li><li>• run code</li><li>• step through the code one line at a time</li><li>• change the speed</li></ul>	
2	2

<p>Ask students to browse to getcoding.io and start moving through the challenges in the Calling Functions squirrel activity. Explain the goal of the activity: Move the squirrel to the nut and pick it up. Tell students they are free to continue on to the second squirrel challenge when they are ready.</p>	<p>Students start solving the challenges in the Calling Functions squirrel activities.</p>
<p><b>3</b> When students get stuck, we suggest using the Read, Write, and Debug protocols to support students. Ask students to imagine being on the Coder team from the group activities. They should try to play the roles of the writer and navigator. Then, ask students to imagine being a Bot to understand how the computer reads the code.</p>	



## PAIR IDEA EXCHANGE



Length: 5 minutes

Talk with a peer about how the code is working.

Prep: None

Teacher Actions	Student Actions
<p>1 Ask students to pause their progress and talk with a neighbor about their current problem. What is their plan? What have they tried? Is there anything standing in their way? Important: Ask students to offer questions instead of solutions.</p>	<p>1 Students pause their progress and talk with a neighbor about the problem they are currently trying to solve.</p>



## RESUME HELPING THE SQUIRREL GATHER ACORNS



Length: 10 minutes

Students continue writing code in JavaScript on the getcoding.io platform to help the squirrel gather acorns.

Prep: None

Teacher Actions	Student Actions
<p>1 Ask students to resume coding individually in the Calling Functions squirrel activity.</p>	<p>1 Students resume coding.</p>



## UNPLUGGED EXIT TICKET



**Length:** 5 minutes

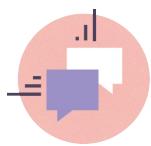
Students individually fill out an exit ticket focused on simple sequence.

**Prep:** Hand out [Lesson 8 | Exit Ticket](#).

Teacher Actions	Student Actions
<p>1 Individual Work: Ask students to fill out the Exit Ticket. Draw their attention to the Elements on the Exit Ticket worksheet because they differ ever so slightly from the squirrel elements.</p>	<p>1 Students fill out the exit ticket.</p>



# Space Ranger and Magic W Plugged



## OVERVIEW

Students write JavaScript online in a series of increasingly difficult challenges involving maneuvering a squirrel to gather nuts.



## OBJECTIVES

- 
1. Students will become proficient at assembling JavaScript commands in sequence.



## AGENDA

---

**Length:** 45 minutes

1. Warm-up
2. Help the Space Ranger
3. Pair Idea Exchange
4. Continue Coding



## MATERIALS

---

1. [Lesson 9 | Warm-up Worksheet](#)
2. Laptops/Computers
3. Scratch paper grids
4. Small turtle cutout for each student
5. Magnetic turtle
6. Scratch paper grids
7. Pencils
8. Whiteboard



## UNPLUGGED WARM-UP



Length: 10 minutes

Practice writing basic JavaScript to maneuver the squirrel to the nut.

Prep: Hand out the [Lesson 9 | Warm-up Worksheet](#).

Teacher Actions	Student Actions
<p><b>1</b> Individual Work: Ask students to write the code to maneuver the squirrel to the nut in the [Lesson 9   Warm-up Worksheet][warm-up]. Consider reminding students of the proper JavaScript syntax (see Elements on the worksheet).</p>	<p><b>1</b> Students individually fill out the Warm-up worksheet.</p>
<p><b>2</b> Draw the squirrel, nut, and grid on the whiteboard and code the solution with the students, randomly calling on one student at a time to provide each next line of code.</p>	<p><b>2</b> If called on, students provide the next line of code.</p>



## HELP THE SPACE RANGER



Length: 15 minutes

Students write code in JavaScript on the getCoding.io platform to help the Space Ranger gather parts. Students are practicing simple sequences with a limited set of elements.

Prep: Students should have their own computers.

Teacher Actions	Student Actions
<p>1 Ask students to browse to getcoding.io and start moving through the challenges in Space Ranger and Space Ranger 2. Explain to the students that this challenge is a bit harder because they can no longer turn right and left. They can only turn left by using the 'rotate' command.</p>	<p>1 Students start solving the challenges in the Space Ranger activities.</p>
<p>2 When students get stuck, we suggest using the Read, Write, and Debug protocols to support students. Ask students to imagine being on the Coder team from the group activities. They should try to play the roles of the writer and navigator. Then, ask students to imagine being a Bot to understand how the computer reads the code.</p>	



## PAIR IDEA EXCHANGE



**Length:** 5 minutes

Talk with a peer about how the code is working.

**Prep:** None

Teacher Actions	Student Actions
<p><b>1</b> Ask students to pause their progress and talk with a neighbor about the problem they are currently trying to solve. What is their plan? What have they tried? Is anything standing in their way? Important: Ask students to offer questions instead of solutions.</p>	<p><b>1</b> Students pause their progress and talk with a neighbor about the problem they are currently trying to solve.</p>



## CONTINUE CODING



Length: 15 minutes

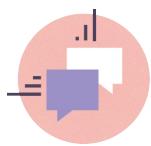
Students continue writing code in JavaScript on the getCoding.io platform for the two Space Ranger activities. When they finish the Space Ranger activities, students should move on to Magic Words. Talk about the rules to Magic Words: Whenever you call out a spell in your program, all of the gates with that spell will move (either up or down, depending on its last position).

Prep: None

Teacher Actions	Student Actions
<p>1 Ask students to resume coding individually.</p>	<p>1 Students resume coding</p>



# Coding arguments Unplugged



## OVERVIEW

Introduce arguments by having kids do a very repetitive task.



## OBJECTIVES

---

1. Students will be able to explain the advantage of using arguments
2. Students will be able to call functions with an argument



## AGENDA

---

**Length: 45 minutes**

1. Warm-up - Large pixel bot grid
2. Arguments Analogies - Explore arguments using golf swing and drill bit analogies.
3. Pixel bot challenge - Solve pixel bot challenges with arguments



## VOCAB

---

**Argument** - Specific value supplied to a function call



## MATERIALS

---

1. Lesson 10 | Warm-up worksheet
2. Lesson 10 | Worksheet 1

3. Lesson 10 | Worksheet 2
4. Laptops/Computers
5. Scratch paper grids
6. Small turtle cutout for each student
7. Magnetic turtle
8. Scratch paper grids
9. Pencils
10. Whiteboard



## WARM-UP



**Length:** 10 minutes

Students solve a puzzle in a large pixelbot grid.

**Prep:**

- Draw the Pixel Bot image from Lesson 10 | Warm-up worksheet on the whiteboard
- Distribute Lesson 10 | Warm-up worksheet

Teacher Actions	Student Actions
<p>1 Individual work: Ask students to write code to create the image from Lesson 10   Warm-up worksheet. Remind students that this exercise is using the icon language that they learned in Lesson 1.</p>	<p>1 Students individually fill out the Lesson 10   Warm-up worksheet</p>
<p>2 Randomly call on one student at a time to provide each next line of code.</p>	<p>2 If called on, students provide the next line of code.</p>
<p>3 Discuss what made this particular picture difficult or frustrating to code?</p> <div data-bbox="303 1462 600 1657" style="background-color: #f0f0f0; padding: 10px;"> <p>Possible answer: It required a lot of code because of the size of the grid.</p> </div>	<p>3 Students raise their hands to provide an answer.</p>



# GOLF SWING AND DRILL BITS



Length: 20 minutes

Explore golf swing and drill bit analogies that help students arrive at concept of parameters/arguments.

Prep: Distribute Lesson 10 | Worksheet 1

Teacher Actions	Student Actions
<p>1 Model a golf swing for students. Show how the same golf swing is used for different clubs. Show the Lesson 10   Worksheet 1 golf diagram on the board and walk students through it.</p>	
<p>2 Model using a drill for students. Show how the same drill motion is used for different drill bits. Show the Lesson 10   Worksheet 1 drill bit diagram on the board and walk students through it.</p>	

<p>3 Point students' attention to Lesson 10   Worksheet 1. Ask students to find what is similar about the two situations depicted.</p>	<p>3 Students look at Lesson 10   Worksheet 1</p>
<p>4 Individual work: Ask students to write down an answer.</p>	<p>4 Students individually write down their answers on the worksheet.</p>
<p>5 With a partner, students discuss their findings.</p>	<p>5 Students get with a partner and discuss their answers.</p>
<p>6 As a whole class, discuss the similarities between the two situations.  Answer: The process is exactly the same (the golf swing never changes; the drill and the drill motion never change), but we can customize the output by changing the inputs (golf club, drill bit).</p>	<p>6 Students raise their hands to offer answers.</p>



# ARGUMENTS



**Length: 5 minutes**

Explain how to use arguments through observation.

**Prep:** None

Teacher Actions	Student Actions
<p><b>1</b> Point students back to the problem on the whiteboard from Lesson 1   Warm-up worksheet.</p>	
<p><b>2</b> Tell students that the elements can use an argument. An argument is extra information to customize the output of a function. The argument goes into the space to the right of the element</p> <div style="background-color: #f0f0f0; padding: 5px; margin-top: 10px;"> Example: → 5 </div>	
<p><b>3</b> Ask students what they think the argument will do in</p>	<p><b>3</b> Students raise their hand to provide an answer.</p>

<p>the case of the arrows?</p> <p>Answer: The number controls how many spaces to move in that direction.</p>	
<p>6 Ask students how adding a number next to the arrow icon relates to changing clubs in the golf swing?</p> <p>Answer: In both examples, the action is the same (swing the golf club, paint the square) but the input can be changed to customize the output.</p>	<p>6 Students raise their hand to provide an answer.</p>
<p>7 Solve the warm up problem while narrating the steps out loud.</p>	<p>7 Students observe as the teacher demonstrates how to solve the problem using arguments.</p>



## CODING WITH ARGUMENTS



Length: 10 minutes

Students use arguments to solve coding challenges.

Prep:

- Distribute Lesson 10 | Worksheet 2

Teacher Actions	Student Actions
<p>1 Individual work: Students work on the problems on Lesson 10   Worksheet 2. Remind students to use arguments to solve the problems more efficiently.</p>	<p>1 Students individually fill in their worksheet.</p>

# Coder Level 3



Students are introduced to core coding concepts: sequences, calling functions, passing arguments, creating loops, and using conditionals. All of the work in this unit is written in fully typed JavaScript. This course includes lesson plans: unplugged and online, worksheets, and a custom-designed coding environment.

## Lessons

- [sequencing pixels JS](#)
- [write some code](#)
- [write read repeat](#)
- [pixel bot online](#)
- [winter is coming - gather food](#)
- [space ranger and magic words](#)
- [coding arguments in JS](#)
- [practicing arguments](#)
- [fire ice and squirrels](#)

## Download lesson plans

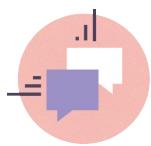
[Download](#)

## Workbook

[Download](#)



# Sequencing Pixels Unplugged



## OVERVIEW

Students program Pixel Bots to paint, focusing on sequence.



## OBJECTIVES

---

- Students will learn that computers run code in a sequence.
- Students will learn how to read, write, and execute code in a sequence.



## AGENDA

---

**Length:** 45 minutes

1. Welcome to coding (10 minutes)
2. Predict pixel bot JS (15 minutes)
3. Explain sequence (10 minutes)
4. Read pixel bot sequence (10 minutes)



## VOCAB

---

- Sequence - The idea that statements must be performed in the order they are written.
- Function call - A programming element that tells the computer to do something. In the beginning, most function calls will cause the computer to perform an action.



## MATERIALS

---

1. [Lesson 1 | Warm-up Worksheet](#)
2. [Worksheet 1: Page 1 & Page 2](#)
3. Small pixel bot cutout for each student
4. Magnetic pixel bot
5. Scratch paper grids
6. Pencils
7. Whiteboard



## WELCOME TO CODING



Length: 10 minutes

Introduce students to the world of coding and get them excited about its endless possibilities.

Prep: Queue up video <http://tinyurl.com/q966xd5>

Teacher Actions	Student Actions
<p>1 Lead a discussion about coding and what it means to be a coder. Suggested script:</p> <p>Starting with this class you are now coders. What do you think it means to be a coder? Where is code used in our world?</p>	<p>1 Students raise their hands to give responses to the questions.</p>
<p>2 Chart student responses on the board.</p>	
<p>3 Fill in additional interesting uses for code on the board, such as autonomous cars, streetlights, music, etc.</p>	
<p>4 Watch video: A day in the life of a software engineer.</p>	



## PREDICT PIXEL BOT JS



**Length:** 15 minutes

Students individually predict the outcome of sequences and then regroup to discuss findings.

**Prep:** Distribute [Lesson 1 | Warm-up Worksheet](#)

Teacher Actions	Student Actions
<p><b>1</b> Tell students: Before we can write code, we need to learn how to read code</p>	
<p><b>2</b> Discuss the elements at the top of [Lesson 1   Warm-up Worksheet][warm-up] and ask students to speculate about what they mean.</p> <p>Answer:</p> <ul style="list-style-type: none"> <li>• <code>up()</code> - move up one square</li> <li>• <code>down()</code> - move down one square</li> <li>• <code>right()</code> - move to the right one square</li> </ul>	<p><b>2</b> Students raise their hands to give answers.</p>

## sequencing pixels JS

<ul style="list-style-type: none"><li>• <code>left()</code> - move to the left one square</li><li>• <code>paint()</code> - paint the square that the pixel bot is on top of</li></ul>	
<p><b>3</b> Individual Work: Tell students to read the elements on the worksheet and paint (color in) the correct square. While students are working on the worksheet, recreate the problems on the board.</p>	<p><b>3</b> Students work individually on their worksheet.</p>
<p><b>4</b> After they are finished, discuss the answers and how the students got to those answers. What is the difference between the two problems? Does the order of the elements matter?</p>	<p><b>4</b> Students raise their hands to give answers.</p>
<p><b>5</b> Students write in what each element means on their worksheets.</p>	<p><b>5</b> Students write in what each element means on their worksheet.</p>





## EXPLAIN SEQUENCE



Length: 10 minutes

Demonstrate how to read code by reading and stepping through three or four example programs.

Prep:

1. Draw a blank 3x3 grid on the whiteboard
2. Write a short (3 line) program on the whiteboard

Teacher Actions	Student Actions
<p>1 Explain that when a computer executes code, it runs it in the order that it is written. This is called sequence.</p>	
<p>2 Explain that these programming elements are part of JavaScript. These particular programming elements are all function calls and that we know they are function calls because they have an open and closed parenthesis after the name.</p>	

<p>3 Point to the program on the whiteboard and ask students, "What is the first line of code?" After they answer, put a number 1 next to the corresponding line. Move the pixel bot according to the line of code just numbered.</p>	<p>3 Students raise their hands to answer questions.</p>
<p>4 Continue reading and stepping one line at a time. Trace the path of the pixel bot as it moves and shade in the squares whenever it paints.</p>	
<p>5 Show students three new examples (design these problems on the fly, making them interesting and complex enough), reading and stepping together as a class.</p>	<p>5 Students follow along and offer answer for what each action does.</p>



## READ PIXEL BOT ICONS



**Length:** 5 minutes

Students individually practice reading code.

**Prep:** Distribute Worksheet 1: [Page 1](#) & [Page 2](#)

Teacher Actions	Student Actions
<p><b>1</b> Individual Work: Leave the worked example from the previous activity on the whiteboard. Ask students to individually fill out the worksheet. Remind students to trace the path of the pixel bot and to shade in squares whenever the pixel bot paints.</p>	<p><b>1</b> Students read the code, trace the pathway of the pixel bot, and paint the correct blocks on the worksheet.</p>

write some code

---

# Write Some Code Unplugged



## OVERVIEW

Students learn the different components of reading and writing code by exploring the roles (writer, reader, navigator, and stepper) of Coders & Bots.



## OBJECTIVES

---

- Students will be able to read basic code.
- Students will be able to write basic code.
- Together as a class, students will be able to enact the Coders & Bots roles of Writer, Navigator, Reader, and Stepper to write and read code.



## AGENDA

---

**Length: 45 mintues**

1. Pixel Bots: Practice Reading Code
2. Pixel Bots: Write Code
3. Pixel Bots: Write Code Together



## VOCAB

---

- Computer - A device that can be instructed to do something.
- Program - A list of statements that a computer can perform.



## MATERIALS

---

1. [Lesson 2 | Warm-up Worksheet](#)
2. [Lesson 2 | Worksheet 1](#)
3. Small pixel bot cutout for each student
4. Magnetic pixel bot
5. Scratch paper grids
6. Pencils
7. Whiteboard



## PIXEL BOTS: PRACTICE READING CODE



Length: 15 minutes

Students practice reading basic code sequences.

Prep: Draw on the whiteboard the grid and the lines of code from the example problem on the front page of [Lesson 2](#) | [Warm-up Worksheet](#).

Teacher Actions	Student Actions
<p>1 Remind students of the value of revisiting ideas explored in previous lessons. Revisiting past ideas helps to see them in a new light and makes sure they are not forgotten.</p>	
<p>2 Step through the example problem as a whole class, talking through the process of reading code.</p>	
<p>3</p>	<p>3</p>

<p>Hand out [Lesson 2   Warm-up Worksheet][warm-up].</p>	<p>Students place their pixel bot at the starting square. Students read the code and move their pixel bot. Students do this a few times to get into a rhythm.</p>
<p>4 Individual Work: Ask students to place the movable pixel bot at the start block of the example problem just demonstrated on the board. Students should work individually, reading the code and moving their pixel bot along. Ask the students to get into a rhythm of reading and stepping.</p>	
<p>5 Individual Work: Ask students to flip the page and perform the same exercise with the new problem. Ask students to shade in the appropriate squares and trace the pathway of the pixel bot.</p>	<p>5 Students turn the page and attempt to solve the problem.</p>

6

On the board, draw the empty grid and the code from the problem students have been working on. Read the code and step the pixel bot (tracing its pathway and shading in squares), narrating your thought process.



## PIXEL BOTS: WRITE CODE



Length: 10 minutes

Students write code to produce a simple pixel bot image.

Prep: Hand out [Lesson 2 | Worksheet 1](#).

Teacher Actions	Student Actions
<p>1 Talk to students about how they have already learned to read code that other people have written. Now they are going to explore writing code.</p>	
<p>2 Individual Work: Have students place their pixel bot at the start square. Ask students to write code that produces the provided pixel bot image. The students should think through their plan for their code, write their code, and move their pixel bot along.</p>	<p>2 Students place their pixel bot at the starting square, write code that creates the picture, and enact the pixel bot actions each step of the way.</p>

3

As a whole class, solicit ideas from multiple students about how they designed their code to solve the problem. Hold off on presenting the correct answer until the next activity.

3

Students share their code, especially if they have solved the problem a different way.



## PIXEL BOTS: WRITE CODE TOGETHER



Length: 20 minutes

Teacher shows student how to write code, emphasizing roles of the Writer, Navigator, Reader, and Stepper (see Coders and Bots Protocol). The students then enact these roles together as a whole class solving a new problem.

Prep: Draw the problem from [Lesson 2 | Worksheet 1](#) on the whiteboard.

Teacher Actions	Student Actions
<p>1 Code the solution to [Lesson 2   Worksheet 1] [worksheet1] on the whiteboard. Voice your code writing process along the way: write and number a new line of code and only then move the pixel bot on the whiteboard. Use this as an opportunity to discuss how there are different ways to solve the same problem. After the code is written, introduce the idea</p>	

that the computer reads code in sequence.

2

After coding the solution, explain that you are switching gears into Bot mode. Read each line and step the pixel bot, checking to see if you coded the correct solution. Add a new problem on the board: a checkerboard pattern. Divide the class in half and follow the process spelled out in the whole class section of the Coders and Bots Protocol. Students start as Coders. Assign half of the students to be writers and half to be navigators; one person from each team walks up to collaboratively add and enact a line of code. The two students then tag in a member of their team to walk up and write the next line of code. Allow the class to code a wrong solution.

2

Students take turns walking up to the board to write and navigate code.

<p>3 The whole class then switches roles to become Bots. Use this opportunity to emphasize the definition of a computer (see vocabulary section above). Following the same procedure as above, ask students to come up to the board to take turns reading and stepping through one line of code at a time before tagging in a new classmate.</p>	<p>3 Students take turns walking up to the board to read and step through the solution.</p>
<p>4 Have students switch back and forth between Coders and Bots until they code the solution on the checkerboard.</p>	<p>4 Students switch back and forth between Coders and Bots to complete the puzzle.</p>
<p>5 Summarize any conceptual difficulties. Introduce and define a Program (see vocabulary section above), and explain how it connects to the code the students just wrote.</p>	

<p>6</p> <p>Ask students to write down on a piece of paper what each of the four roles does, focusing on one role at a time. After each role definition, ask students to share their definition with the whole class. Pool the students' ideas into overall definitions of the roles.</p>	<p>6</p> <p>Students provide descriptions of what each role entails.</p>



# Write.Read.Repeat. Unplugged



## OVERVIEW

Students continue to solve pixel bot problems by playing Coders & Bots to read and write code.



## OBJECTIVES

---

- Students will perform the Coder & Bots roles of Writer, Navigator, Reader, and Stepper
- Students will collaborate in small groups to write and read code
- Students will become more proficient at writing code



## AGENDA

---

Length: 45 minutes

1. Pixel Bots: Write Code Warm-up
2. Pixel Bots: Coders and Bots
3. Pixel Bots: Write Code Exit Ticket



## VOCAB

---

- Programming Element - A command the computer understands.
- Action - An observable event that the code produces
- Error - The program fails to run because the computer cannot execute the code



## MATERIALS

---

1. Lesson 3 | Warm-up Worksheet
2. Lesson 3 | Worksheet 1
3. Lesson 3 | Worksheet 2
4. Lesson 3 | Worksheet 3
5. Lesson 3 | Worksheet 4
6. Lesson 3 | Exit Ticket Worksheet
7. Scratch paper grids
8. Small pixel bot cutout for each student
9. Magnetic pixel bot
10. Scratch paper grids
11. Pencils
12. Whiteboard



## CLASSROOM SETUP

---

Pods of four.



# PIXEL BOTS: WRITE CODE WARM-UP



**Length:** 15 minutes

Students write code for a simple pixel bot image.

**Prep:** Hand out Lesson 3 | Warm-up.

Teacher Actions	Student Actions
<p><b>1</b> Individual Work: Ask students to solve the problem on [Lesson 3   Warm-up Worksheet][warm-up].</p>	<p><b>1</b> Students individually solve the problem on the worksheet.</p>
<p><b>2</b> As students attempt the problem, draw the problem on the whiteboard.</p>	
<p><b>3</b> Code the solution to the problem as a whole class. Call on students at random to provide each next line of code. Each student should walk up to the board and write the</p>	<p><b>3</b> If called on, students walk up to the board to write in (or read) the next line of code.</p>

next line of code. With each new line of code, you should play the role of Navigator, moving the turtle. Call out the roles of Writer and Navigator to make them explicit. On occasion, pick a Reader and Stepper to walk up to the board to test the code starting from line one.



## PIXEL BOTS: CODERS AND BOTS



**Length:** 30 minutes

Students work in groups to write and read code to produce two pixel bot images.

**Prep:** Consider having paper bags for each group. Each paper bags contains the four roles (Writer, Navigator, Reader, Stepper).

Teacher Actions	Student Actions
<p>1 Teacher breaks students into groups of four and randomly assigns roles for the Programming Team and the Computer Team. (Consider handing each group a paper bag with the four roles inside – students reach into the bag and grab a role.) In this first round, groups either get [Lesson 3   Worksheet 1] [worksheet1] or [Lesson 3   Worksheet 2] [worksheet2] (alternate between groups). Bots should always help the Coders during</p>	<p>1 Students enact their Coders and Bots roles. They write code to create the pixel bot image.</p>

the code writing phase of the activity.

2

Follow the Coders and Bots Protocol by having the Bot Team switch to a new group when it comes time to check the code. Make sure that the Coders fold back their paper to hide the provided pixel bot image before the Bots arrive (ensuring that the Bots are not biased in their reading). The Bots should test the code on an empty scratch paper grid. Ask students, "What do you think the computer does when it cannot understand the code or the code forces the turtle to break the rules (go outside the grid)?" Answer: An error! If the Bots notice this happening, they should report the error to the programming team and tell them what line the error happened on.

2

The Bots switch to a new group to assess code. If the Bots find an error, report it to the Coders and tell them what happened and the line number the error happened on.

<p>3 After one round, pass out [Lesson 3   Worksheet 3] [worksheet3] or [Lesson 3   Worksheet 4] [worksheet4] to the groups (alternate again) and repeat the Coders and Bots Protocol.</p>	<p>3 Students repeat the above process with new coding challenges.</p>	
<p>4 In whole class mode, ask students to share out any disagreements they had in the write and read process. Use this as an occasion to firm up any misconceptions. Also use this time to introduce and define Elements and Actions (see vocabulary section above).</p>	<p>4 Students summarize and describe the disagreements they could not resolve.</p>	



## PIXEL BOTS: WRITE CODE EXIT TICKET



**Length:** 5 minutes

Time permitting, students individually fill out an exit ticket to check for understanding.

**Prep:** Hand out [Lesson 3 | Exit Ticket Worksheet](#)

Teacher Actions	Student Actions
<p>1 Ask students to individually work on completing the [Lesson 3   Exit Ticket Worksheet] [wrap-up].</p>	<p>1 Students work individually on the [Lesson 3   Exit Ticket Worksheet] [wrap-up].</p>



Pixel Bots Online  
Plugged



## OVERVIEW

Students review writing code offline. Then, students demonstrate their learning by writing programs to complete online pixel bot challenges.



## OBJECTIVES

---

- Students will write programs using JavaScript.
- Students will continue to develop proficiency in writing and reading code.



## AGENDA

---

**Length: 45 mintues**

1. Pixel Bots: Warm-up
2. Introduce Pixel Bots Online
3. Pixel Bots Online Practice
4. Pixel Bots: Exit Ticket



## VOCAB

---

- Program - A list of statements that a computer can perform.



## MATERIALS

---

1. Lesson 4 | Worksheet 1

2. [Lesson 4 | Exit Ticket Worksheet](#)
3. Scratch paper grids
4. Small pixel bot cutout for each student
5. Magnetic pixel bot
6. Scratch paper grids
7. Pencils
8. Whiteboard



## PIXEL BOTS: WARM UP



Length: 10 minutes

Create a medium difficulty pixel bot image for the students on the whiteboard. Students solve the problem individually and then check the work of a peer.

Prep: Create a medium difficulty pixel bot image on the whiteboard.

Teacher Actions	Student Actions
<p>1 Individual Work: Ask students to get a blank piece of paper, write line numbers, and write the code that creates the pixel bot image on the board.</p>	<p>1 Students work individually on coding a solution to the challenge.</p>
<p>2 Students discuss the coding solution with a peer.</p>	<p>2 When they finish, students check their work with a partner after both have finished.</p>



# INTRODUCE PIXEL BOT ONLINE



Length: 10 minutes

Show students how to solve a problem on pixel bot online. Also revisit **program** as a vocabulary word.

Prep: Browse to [www.pixelbots.io](http://www.pixelbots.io) and project onto the wall.

Teacher Actions	Student Actions
<p>1 Now that students have practice creating code sequences on paper, they are ready to start writing programs.</p>	<p>1 Add student actions here and match numbers to teacher actions</p>
<p>2 Revisit the idea that a program is a sequence that a computer is able to understand.</p>	
<p>3 On a projector, show students <a href="http://www.pixelbots.io">www.pixelbots.io</a></p>	
<p>4</p>	

Use the problem you created in the warm-up to demonstrate how to solve the problem in the online interface. Show students a variety of features in the pixel bot interface:

- How to add code
- How to delete code
- How to run program
- How to reset after run
- How to insert code

5

Ask students, "What do you think we should do if we get stuck on a problem?"

5

Answer: Step through code starting at the beginning like you do as a Bot.



## PIXEL BOT ONLINE PRACTICE



Length: 20 minutes

Students are given a set of pixel bot images to reproduce on the computer. Partners star the ones that are completed. Remind students that the images with lots of shaded squares will be difficult, but they should remember to read code from the beginning when things go wrong.

Prep: Distribute the [Lesson 4 | Worksheet 1](#) worksheet and write [www.pixelbots.io](http://www.pixelbots.io) up on the board.

Teacher Actions	Student Actions
<p>1 Explain the exercise:</p> <ul style="list-style-type: none"><li>• The goal is to recreate each of the images from the worksheet by creating a program on the website.</li><li>• Tell students that they will act as testers for the person sitting next to them:<ul style="list-style-type: none"><li>◦ When they finish writing a program, students have their</li></ul></li></ul>	<p>1 Students are faced away from their computers toward the teacher.</p>

partner check to make sure the images match up. If they are a match, the tester puts a checkmark next to the image.

- o The programmer should then explain how their code works to the tester. If the tester is satisfied with the explanation, the tester checks the explain box.
- o The programmer can now continue on to the next challenge.

2

Students work on completing the challenges.

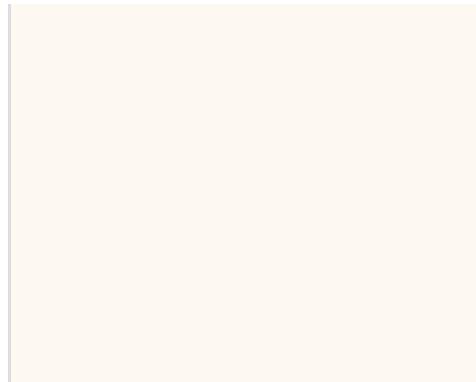
2

Students get on their computers and go to [www.pixelbots.io](http://www.pixelbots.io). Students work

	<p>individually on recreating each of the images with code.</p> <ul style="list-style-type: none"><li>When they finish a puzzle, students have their partner (student sitting next to them) check to make sure the images match up. If they are a match, the checker puts a checkmark next to the image and the programmer can continue on to the next challenge.</li></ul>
<p><b>3</b></p> <p>When students get stuck, ask them to imagine being on the Coder team from the group activities. They should try to play the roles of the writer and navigator. Then, ask students to imagine being a Bot to understand how the computer is reading the code.</p>	<p><b>3</b></p> <p>Students raise their hands to provide answers.</p>

4

Discuss: Which was the hardest coding challenge? Why? Was there more than one way to solve the problem?





## PIXEL BOTS: EXIT TICKET



**Length:** 5 minutes

Students complete Exit Ticket.

**Prep:** Distribute [Lesson 4 | Exit Ticket Worksheet](#).

Teacher Actions	Student Actions
<p><b>1</b> Individual Work: Tell students they are going to work individually on coding the answer on the worksheet.</p>	<p><b>1</b> Students write their answers on the worksheet.</p>



Gather Food - Winter is Coming  
plugged



## OVERVIEW

Students write JavaScript on getcoding.io to solve increasingly difficult challenges involving moving a squirrel to gather nuts.



## OBJECTIVES

---

1. Students write basic JavaScript to solve simple navigation problems.



## AGENDA

---

**Length:** 45 minutes

1. Warm-up
2. Help the Squirrel Gather Acorns
3. Pair Idea Exchange
4. Resume Helping the Squirrel Gather Acorns
5. Unplugged Exit Ticket



## VOCAB

---

**Code Editor** - The place where coders assemble their program.



## MATERIALS

---

1. [Lesson 5 | Warm-up Worksheet](#)
2. [Lesson 5 | Exit Ticket](#)

3. Laptops/Computers
4. Scratch paper grids
5. Small turtle cutout for each student
6. Magnetic turtle
7. Scratch paper grids
8. Pencils
9. Whiteboard



## WARM-UP

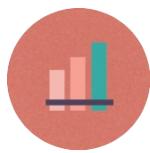


Length: 10 minutes

Students practice writing basic javascript to create a simple Pixel Bot drawing.

Prep: Hand out [Lesson 5 | Warm-up Worksheet](#)

Teacher Actions	Student Actions
<p><b>1</b> Individual Work: Ask students to write the code to produce the Pixel Bot image in the [Lesson 5   Warm-up Worksheet] [warm-up]. Consider reminding students of the proper JavaScript syntax (see Elements on the worksheet).</p>	<p><b>1</b> Students individually fill out the Warm-up Worksheet.</p>
<p><b>2</b> Draw the Pixel Bot image on the whiteboard and code the solution with the students, randomly calling on one student at a time to provide each next line of code. (Note the problem can be solved in different ways. Students should follow the class' ongoing code which may differ from their own solution).</p>	<p><b>2</b> If called on, students call out the next line of code.</p>



## HELP THE SQUIRREL GATHER ACORNS



Length: 15 minutes

Students write code in JavaScript on the getcoding.io platform to help the squirrel gather acorns. Students are practicing simple sequences.

**Prep:** Have the getcoding.io platform open on your browser and projected on the wall. Students should also have their own computers.

Teacher Actions	Student Actions
<p>1 Walk students through the getcoding.io platform. Show students how to:</p> <ul style="list-style-type: none"><li>• open activities</li><li>• use the Code Editor (where to type)</li><li>• see elements</li><li>• run code</li><li>• step through the code one line at a time</li><li>• change the speed</li></ul>	
2	2

<p>Ask students to browse to getcoding.io and start moving through the challenges in the Calling Functions squirrel activity. Explain the goal of the activity: Move the squirrel to the nut and pick it up. Tell students they are free to continue on to the second squirrel challenge when they are ready.</p>	<p>Students start solving the challenges in the Calling Functions squirrel activities.</p>
<p><b>3</b> When students get stuck, we suggest using the Read, Write, and Debug protocols to support students. Ask students to imagine being on the Coder team from the group activities. They should try to play the roles of the writer and navigator. Then, ask students to imagine being a Bot to understand how the computer reads the code.</p>	



## PAIR IDEA EXCHANGE



**Length:** 5 minutes

Talk with a peer about how the code is working.

**Prep:** None

Teacher Actions	Student Actions
<p><b>1</b> Ask students to pause their progress and talk with a neighbor about their current problem. What is their plan? What have they tried? Is there anything standing in their way? Important: Ask students to offer questions instead of solutions.</p>	<p><b>1</b> Students pause their progress and talk with a neighbor about the problem they are currently trying to solve.</p>



## RESUME HELPING THE SQUIRREL GATHER ACORNS



Length: 10 minutes

Students continue writing code in JavaScript on the getcoding.io platform to help the squirrel gather acorns.

Prep: None

Teacher Actions	Student Actions
<p>1 Ask students to resume coding individually in the Calling Functions squirrel activity.</p>	<p>1 Students resume coding.</p>



## UNPLUGGED EXIT TICKET



**Length:** 5 minutes

Students individually fill out an exit ticket focused on simple sequence.

**Prep:** Hand out [Lesson 5 | Exit Ticket](#).

Teacher Actions	Student Actions
<p>1 Individual Work: Ask students to fill out the Exit Ticket. Draw their attention to the Elements on the Exit Ticket worksheet because they differ ever so slightly from the squirrel elements.</p>	<p>1 Students fill out the exit ticket.</p>

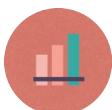


# Space Ranger and Magic W Plugged



## OVERVIEW

Students write JavaScript online in a series of increasingly difficult challenges involving maneuvering a squirrel to gather nuts.



## OBJECTIVES

---

1. Students will become proficient at assembling JavaScript commands in sequence.



## AGENDA

---

**Length:** 45 minutes

1. Warm-up
2. Help the Space Ranger
3. Pair Idea Exchange
4. Continue Coding



## MATERIALS

---

1. [Lesson 6 | Warm-up Worksheet](#)
2. Laptops/Computers
3. Scratch paper grids
4. Small turtle cutout for each student
5. Magnetic turtle
6. Scratch paper grids
7. Pencils
8. Whiteboard



## UNPLUGGED WARM-UP



Length: 10 minutes

Practice writing basic JavaScript to maneuver the squirrel to the nut.

Prep: Hand out the [Lesson 6 | Warm-up Worksheet](#).

Teacher Actions	Student Actions
<p><b>1</b> Individual Work: Ask students to write the code to maneuver the squirrel to the nut in the [Lesson 6   Warm-up Worksheet][warm-up]. Consider reminding students of the proper JavaScript syntax (see Elements on the worksheet).</p>	<p><b>1</b> Students individually fill out the Warm-up worksheet.</p>
<p><b>2</b> Draw the squirrel, nut, and grid on the whiteboard and code the solution with the students, randomly calling on one student at a time to provide each next line of code.</p>	<p><b>2</b> If called on, students provide the next line of code.</p>



## HELP THE SPACE RANGER



**Length:** 15 minutes

Students write code in JavaScript on the getcoding.io platform to help the Space Ranger gather parts. Students are practicing simple sequences with a limited set of elements.

**Prep:** Students should have their own computers.

Teacher Actions	Student Actions
<p>1 Ask students to browse to getcoding.io and start moving through the challenges in Space Ranger and Space Ranger 2. Explain to the students that this challenge is a bit harder because they can no longer turn right and left. They can only turn left by using the 'rotate' command.</p>	<p>1 Students start solving the challenges in the Space Ranger activities.</p>
<p>2 When students get stuck, we suggest using the Read, Write, and Debug protocols to support students. Ask students to imagine being on the Coder team from the group activities. They should try to play the roles of the writer and navigator. Then, ask students to imagine being a Bot to understand how the computer reads the code.</p>	



## PAIR IDEA EXCHANGE



Length: 5 minutes

Talk with a peer about how the code is working.

Prep: None

Teacher Actions	Student Actions
<p>1 Ask students to pause their progress and talk with a neighbor about the problem they are currently trying to solve. What is their plan? What have they tried? Is anything standing in their way? Important: Ask students to offer questions instead of solutions.</p>	<p>1 Students pause their progress and talk with a neighbor about the problem they are currently trying to solve.</p>



## CONTINUE CODING



Length: 15 minutes

Students continue writing code in JavaScript on the getCoding.io platform for the two Space Ranger activities. When they finish the Space Ranger activities, students should move on to Magic Words. Talk about the rules to Magic Words: Whenever you call out a spell in your program, all of the gates with that spell will move (either up or down, depending on its last position).

Prep: None

Teacher Actions	Student Actions
<p>1 Ask students to resume coding individually.</p>	<p>1 Students resume coding</p>

# Coding arguments Unplugged



## OVERVIEW

Introduce arguments by having kids do a repetitive task.



## OBJECTIVES

---

1. Students will be able to explain the advantage of using arguments
2. Students will be able to call functions with an argument



## AGENDA

---

**Length: 45 minutes**

1. Warm-up - Large pixel bot grid
2. Arguments Analogies - Explore arguments using golf swing and drill bit analogies.
3. Pixel bot challenge - Solve pixel bot challenges with arguments



## VOCAB

---

**Argument** - Specific value supplied to a function call



## MATERIALS

---

1. [Lesson 7 | Warm-up worksheet](#)
2. [Lesson 7 | Worksheet 1-1](#)

3. [Lesson 7 | Worksheet 1-2](#)
4. [Lesson 7 | Worksheet 1-3](#)
5. [Lesson 7 | Worksheet 2](#)
6. Scratch paper grids
7. Small pixel bot cutout for each student
8. Magnetic pixel bot
9. Scratch paper grids
10. Pencils
11. Whiteboard



## WARM-UP



**Length:** 10 minutes

Students solve a puzzle in a large pixelbot grid.

**Prep:**

- Draw the Pixel Bot image from [Lesson 7 | Warm-up worksheet](#) on the whiteboard
- Distribute [Lesson 7 | Warm-up worksheet](#)

Teacher Actions	Student Actions
<p><b>1</b> Individual work: Ask students to write code to create the image from [Lesson 7   Warm-up worksheet] [warm-up].</p>	<p><b>1</b> Students individually fill out the [Lesson 7   Warm-up worksheet][warm-up]</p>
<p><b>2</b> Randomly call on one student at a time to provide each next line of code.</p>	<p><b>2</b> If called on, students provide the next line of code.</p>
<p><b>3</b> Discuss what made this particular picture difficult or frustrating to code.</p> <div data-bbox="303 1291 600 1484" style="background-color: #f0f0f0; padding: 10px;"> <p>Possible answer: It required a lot of code because of the size of the grid.</p> </div>	<p><b>3</b> Students raise their hands to provide an answer.</p>



# GOLF SWING AND DRILL BITS



**Length:** 20 minutes

Explore golf swing and drill bit analogies that help students arrive at concept of parameters/arguments.

**Prep:** Distribute [Lesson 7 | Worksheet 1-1](#)

Teacher Actions	Student Actions
<p><b>1</b> Individual work: Ask students to fill out [Lesson 7   Worksheet 1-1] [worksheet1-1].</p>	<p><b>1</b> Students fill out Lesson 7   Worksheet 1-1.</p>
<p><b>2</b> As a whole class, pool students' ideas.</p>	<p><b>2</b> Students share ideas.</p>
<p><b>3</b> Individual work: Hand out [Lesson 7   Worksheet 1-2] [worksheet1-2] and ask students to give it a try.</p>	<p><b>3</b> Students fill out [Lesson 7   Worksheet 1-2] [worksheet1-2].</p>
<p><b>4</b> Discuss how to write the proper syntax for the golf and drill bit programs. Write</p>	<p><b>4</b> Students predict the teacher's code.</p>

a few examples of the syntax on the board and ask students to predict how far the ball would go or how big the hole would be.

5

Individual work:  
Hand out the next [Lesson 7 | Worksheet 1-3] [worksheet1-3]  
Cont'd and ask students to map these ideas over to pixel bot.

5

Students fill out next part of [Lesson 7 | Worksheet 1-3] [worksheet1-3].

6

Discuss students' ideas for Question 6. Answer: The process is exactly the same (the golf swing never changes; the drill and the drill motion never change), but we can customize the output by changing the inputs (golf club, drill bit).

6

Students raise their hands to provide answers



# ARGUMENTS



**Length: 5 minutes**

Explain how to use arguments through observation.

**Prep: None**

Teacher Actions	Student Actions
<p>1 Point students back to the problem on the whiteboard from Lesson 1   Warm-up worksheet.</p>	
<p>2 Tell students that the elements can use an argument. An argument is extra information to customize the output of a function. The argument goes in between the parenthesis that follow the name of the function.</p> <p>Example: <code>up( 5 )</code></p>	
3	3

<p>Ask students to say once again what the argument will do in the case of the movements?</p> <p>Answer: The number controls how many spaces to move in that direction.</p>	<p>Students raise their hand to provide an answer.</p>
<p>4 Add <code>paint('blue')</code></p>	
<p>5 Ask students what they think the argument will do in the case of the paint icon?</p> <p>Answer: the argument controls what color the turtle will paint</p>	<p>5 Students raise their hand to provide an answer.</p>
<p>6 Ask students how changing the color next to the icon relates to changing clubs in the golf swing? Answer: In both examples, the action is the same (swing the golf club, paint the square) but the input can be</p>	<p>6 Students raise their hand to provide an answer.</p>

<p>changed to customize the output.</p>	
<p>7 Solve the warm up problem while narrating the steps out loud.</p>	<p>7 Students observe as the teacher demonstrates how to solve the problem using arguments.</p>



## CODING WITH ARGUMENTS



Length: 10 minutes

Students use arguments to solve coding challenges.

Prep:

- Distribute [Lesson 7 | Worksheet 2](#)

Teacher Actions	Student Actions
<p>1 Individual work: Students work on the problems on [Lesson 7   Worksheet 2] [worksheet2]. Remind students to use arguments to solve the problems more efficiently.</p>	<p>1 Students individually fill in the problems on their worksheet.</p>

## Practicing arguments Plugged



## OVERVIEW

Students practice utilizing function calls with arguments offline in coders and bots, and then on pixelbots.io.



## OBJECTIVES

---

1. Students will be able to use arguments to solve programming challenges.
2. Students will become proficient at assembling JavaScript commands in sequence.



## AGENDA

---

**Length: 45 minutes**

1. Warm-up with large pixel bot exercise (5 minutes)
2. Coders and Bots with arguments (30 minutes)
3. Individual work (10 minutes)



## MATERIALS

---

1. [Lesson 8 | Warm-up Worksheet](#)
2. [Lesson 8 | Worksheet 1](#)
3. [Lesson 8 | Worksheet 2](#)
4. [Lesson 8 | Worksheet 3](#)
5. Laptops/Computers
6. Scratch paper grids
7. Small turtle cutout for each student
8. Magnetic turtle

9. Scratch paper grids
10. Pencils
11. Whiteboard



## ACTIVITY TITLE



**Length:** 5 minutes

Students warm up by completing exercise on [Lesson 8 | Warm-up Worksheet](#)

Prep: Distribute [Lesson 8 | Warm-up Worksheet](#)

Teacher Actions	Student Actions
<p><b>1</b> Individual Work: Ask students to write the code to produce the Pixel Bot image in the [Lesson 8   Warm-up Worksheet] [warm-up]. Consider reminding students of the proper JavaScript syntax (see Elements on the worksheet).</p>	<p><b>1</b> Students individually fill out the Warm-up Worksheet.</p>
<p><b>2</b> Draw the Pixel Bot image on the whiteboard and code the solution with the students, randomly calling on one student at a time to provide each next line of code. (Note the problem can be solved in different ways. Students should follow the class' ongoing code which may differ from their own solution).</p>	<p><b>2</b> If called on, students call out the next line of code.</p>



## CODERS AND BOTS



Length: 30 minutes

Students explore more complex coding problems in groups using the coder and bots protocol.

Prep: Distribute [Lesson 8 | Worksheet 1](#)

Teacher Actions	Student Actions
<p>1 Teacher breaks students into groups of four and randomly assigns roles for the Programming Team and the Computer Team. (Consider handing each group a paper bag with the four roles inside – students reach into the bag and grab a role.) In this first round, groups either get [Lesson 8   Worksheet 1] [worksheet1]. Bots should always help the Coders during the code writing phase of the activity.</p>	<p>1 Students enact their Coders and Bots roles. They write code to create the pixel bot image.</p>
<p>2</p>	

Explain that this problem can be solved more than one way. The goal is for students to develop code that solves the problem using the FEWEST lines of code.

3 Follow the Coders and Bots Protocol by having the Bot Team switch to a new group when it comes time to check the code. Make sure that the Coders fold back their paper to hide the provided pixel bot image before the Bots arrive (ensuring that the Bots are not biased in their reading). The Bots should test the code on an empty scratch paper grid. If the Bots notice an error, they should report the error to the programming team and tell them what line the error happened on.

3 The Bots switch to a new group to assess code. If the Bots find an error, report it to the Coders and tell them what happened and the line number the error happened on.

4

4

Discuss as a whole class how many lines it took to complete the image. Then, the group with the fewest lines shares writes their code on the whiteboard. In their groups, students have 2 minutes to discuss the optimal solution. How was this program able to use less lines of code?

Students discuss the code and develop new strategies for the next round.

5 After one round, pass out [Lesson 8 | Worksheet 2] [worksheet2] to the groups and repeat the Coders and Bots Protocol.

5 Students repeat the above process with new coding challenges.

6 In whole class mode, ask students to share out any disagreements they had in the write and read process. Use this as an occasion to firm up any misconceptions.

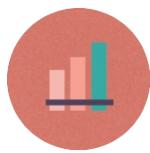
6 Students summarize and describe the disagreements they could not resolve.

7 What strategies did the groups come up with to write the

7 Students raise their hands to offer solutions.

least lines of code?

Answer: The best strategy is to make sure that each move takes the pixel bot to a square that needs to be painted.



## ONLINE ARGUMENTS



Length: 10 minutes

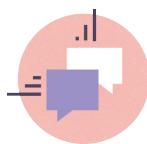
Students practice using arguments to create images on pixelbots.io

Prep: Distribute Lesson 8 | Worksheet 3

Teacher Actions	Student Actions
<p>1 Individual work: Students work on the problems on Lesson 8   Worksheet 3 on pixelbots.io. Remind students to use arguments to solve the problems more efficiently.</p>	<p>1 Students get on their computer and navigate to pixelbots.io to complete the challenges on Lesson 8   Worksheet 3</p>



# Fire Ice and Squirrels Plugged



## OVERVIEW

Students write JavaScript online in a series of increasingly difficult challenges involving maneuvering a squirrel to gather acorns, and a wizard using magic words.



## OBJECTIVES

---

1. Students will become proficient at assembling JavaScript commands in sequence.
2. Students will become proficient at using arguments
3. Students will be able to define a string
4. Students will be able to pass a string as an argument



## AGENDA

---

**Length: 45 minutes**

1. Warm-up
2. Help the squirrel
3. Introduce strings
4. Fire and ice



## VOCAB

---

- String - A sequence of characters or words.



## MATERIALS

---

1. [Lesson 9 | Unplugged Warm-up Worksheet](#)
2. Laptops/Computers
3. Scratch paper grids
4. Small turtle cutout for each student
5. Magnetic turtle
6. Scratch paper grids
7. Pencils
8. Whiteboard



## UNPLUGGED WARM-UP



Length: 10 minutes

Practice writing function calls with arguments to maneuver the pixel bot.

Prep: Hand out the [Lesson 9 | Unplugged Warm-up Worksheet](#).

Teacher Actions	Student Actions
<p>1 Individual Work: Ask students to write the code to maneuver the squirrel to the nut in the [Lesson 9   Unplugged Warm-up Worksheet] [warm-up]. Consider reminding students of the proper JavaScript syntax (see Elements on the worksheet).</p>	<p>1 Students individually fill out the Warm-up worksheet.</p>
<p>2 Draw the example on the whiteboard and code the solution with the students, randomly calling on one student at a time to provide each next line of code.</p>	<p>2 If called on, students provide the next line of code.</p>



## HELP THE SQUIRREL



**Length:** 15 minutes

**Students write code in JavaScript on the getCoding.io platform to help the squirrel gather the acorns. Students are practicing simple sequences with arguments.**

**Prep:** Students should have their own computers.

Teacher Actions	Student Actions
<p>1 Ask students to browse to getcoding.io and start moving through the challenges in Passing Arguments: Squirrel Climber. Explain to the students that now the squirrel movements can take one argument - the number of squares to move.</p>	<p>1 Students start solving the challenges in the Space Ranger activities.</p>
<p>2 When students get stuck, we suggest using the Read, Write, and Debug protocols to support students. Ask students to imagine being on the Coder team from the group activities. They should try to play the roles of the writer and navigator. Then, ask students to imagine being a Bot to understand how the computer reads the code.</p>	



# INTRODUCING STRINGS



**Length:** 5 minutes

Gather students and introduce the concept of strings.

**Prep:**

- Set up projector to show Passing Arguments: Magic Words on getcoding.io
- Write examples of all of the function calls that students have used with arguments on the whiteboard

From pixelbots.io:

```
up(2)  
down(3)  
left(2)  
right(4)
```

From getcoding.io:

```
move(4)
```

Teacher Actions	Student Actions
<p>1 Show students the function calls on the whiteboard. Ask students what pattern they notice about the arguments they have been using.</p> <p>Answer: All of the arguments up until this point have been numbers.</p>	<p>1 Students raise their hands to offer an answer.</p>

<p>2 Show students Passing Arguments: Magic Words. Ask students what kind of argument is needed to cast a spell in the game.</p> <p>Answer: A word</p>	<p>2 Students raise their hands to offer an answer.</p>
<p>3 Explain that in programming text is a different type of data called a string. To let the computer know that a value is a string, it needs to be put inside quotation marks.</p> <p>Ex. <code>cast('fire')</code></p>	
<p>4 As a class do the first puzzle of Passing Arguments: Magic Words. Make sure to draw attention to the quotation marks that go around the string.</p>	



## MAGIC WORDS



Length: 15 minutes

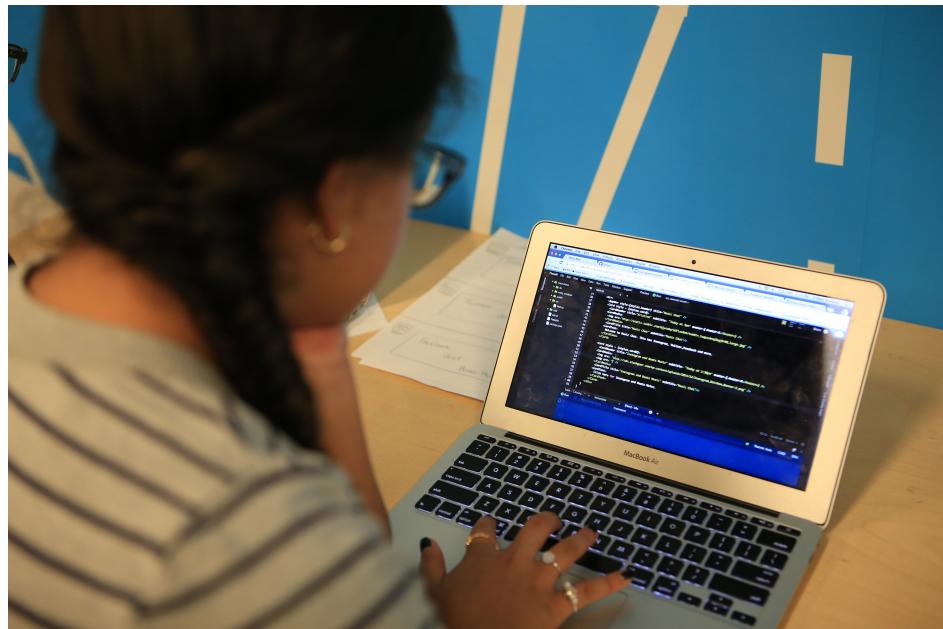
Students continue writing code in JavaScript on the getCoding.io platform for the Passing Arguments: Magic Words. Remind them that in this version, in addition to opening the gates, whenever there is a fire or ice wall, they need to cast a spell of the opposite element.

Prep: None

Teacher Actions	Student Actions
<p>1 Ask students to resume coding individually.</p>	<p>1 Students resume coding</p>



# Developer



The developer series features project and product based curricula designed to introduce and reinforce applied learning in projects motivated by the product design cycle. The lessons in these units are couched in the context of an overarching product that the student is working on developing. Students will create games, apps, and collaborative worlds.

# Curriculum

- [Level 1](#)
- **Level 2 (Coming Soon!)**

## Overview: Escape the Maze

In this 27-lesson course, students master computational thinking skills and the fundamentals of coding to build an interactive maze game with multiple obstacles, enemies, and levels. These lessons incorporate unplugged and online activities that build connections between coding concepts and apply them to projects. Along with hard skills in coding, students learn the processes and mindsets of a computer programmer. Students learn to approach failure as an opportunity, collaborate with peers on writing and reviewing code, and tap into their own creativity without reservations. At the end of the course students apply their knowledge of the software design process to plan an unplugged event - a class-wide Arcade Day - in which they share their final maze projects and celebrate their development as a coder.

# Purpose

1. Empower students to use computational practices to analyze problems, build solutions, and be creative.
2. Empower students to apply computational practices to understand and change the world.

## Big Goals

1. Students will utilize computational thinking to read their world and will utilize their coding skills to write their world.
2. Students will experience failure as a learning opportunity to master core concepts and the process of creation.
3. Students will collaboratively build software and assess its effectiveness for users.

## Big Ideas

1. Software is built by composing simple, common coding building blocks.
2. Reusable and robust software is readable, modular, and testable.
3. Impactful software is built iteratively and collaboratively through stages of planning, enacting, and assessing.
4. When code breaks, we fearlessly, creatively, and systematically debug it.

## Essential Questions

1. How can we use coding building blocks to design something new or break something down?
2. When your code breaks or when you get stuck, what can you do?
3. How do you write code that other programmers can use?
4. How do you adapt your software to meet the needs of your users?

## Developer Level 1



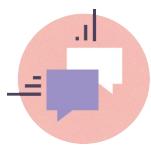
Students begin applying core coding concepts into an Escape the Maze project. In this level, students are introduced to the Scratch block-based language and the offline maze grid. These platforms help student experience and understand new coding concepts such as sequencing, creating loops, and using conditionals.

## Lessons

- [Lesson 1: I am a coder](#)
- [Lesson 2: Getting Started on Scratch](#)
- [Lesson 3: Maze Scavenger Hunt](#)
- [Lesson 4: Dance Off](#)
- [Lesson 5: In the Loop](#)



## Lesson 1: I am a Coder Unplugged



## OVERVIEW

In this lesson, students discover the importance of code in today's world. Additionally, students will repeatedly encounter failure in a positive way through an interactive problem-solving game. They should begin to view failure as a step towards succeeding.

# Print PDF

[Print this lesson](#)



## AGENDA

---

- Do Now: Students write their names and their career (5 min)
- Attention Getting Signal: Teach or review your signal to move from small group to whole group (1-5 min)
- River Crossing Activity: Students repeatedly encounter failure and connect it to progress as they solve the river crossing challenge (25-30 min)
  - Introduce Challenge: Students learn what code is and understand the challenge they will be solving (5 min)
  - Small Groups: Students work in groups to solve the puzzle, returning whole class every few minutes to troubleshoot together and identify their progress through failed solutions.
- Norm Building: Students reflect on how it feels to fail and create a document that outlines how they will support themselves and each other when they encounter failure in the classroom (10 min)



## VOCAB

---

**Code:** A set of instructions designed to be carried out by a computer



## MATERIALS

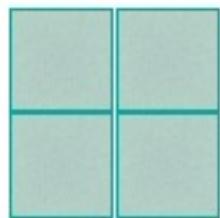
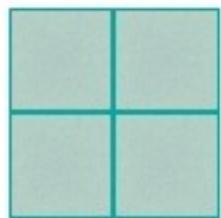
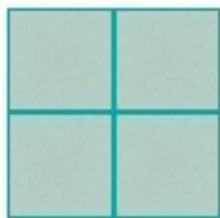
---

- Index Cards (class sets)
- Markers for index cards (class set)
- Code Cards (1 set for each group)

- Felt Strips (1 for each group)
- Paper River Crossing Worksheet (class set)
- Characters (1 set for each group)
- Classroom river (teal foam tiles)
- Teacher magnetic code cards & step arrow
- Teacher Magnetic Characters
- Chart paper (2 pieces)
- Sticky notes, 2 colors (class set of each color)
- Group Roles Sheet

# Ideal Classroom Setup

DESKS IN GROUPS OF FOUR



RIVER MADE OF BLUE FOAM TILES



# Ideal Board Setup

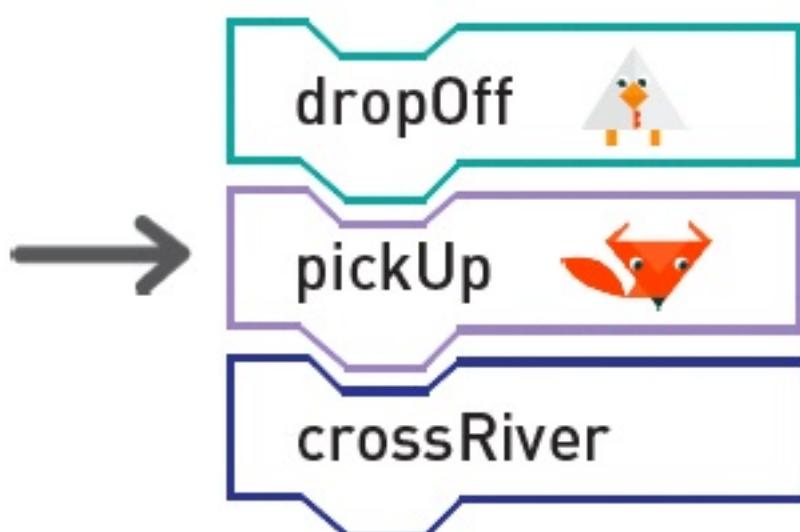
CHART FOR PROGRESS

Debugging Moments	Lines of Code

RIVER DRAWN ON BOARD



MAGNETIC CODE STRIPS, EXECUTION  
ARROW AND CLIPS FOR HANGING



# Small Group Roles

When working in a group each student will have one of these roles assigned by the teacher. These roles will remain consistent throughout the school year, though students should be assigned different roles for different lessons.

Role	Description	Tools
Stepper	Keeps track of what the code has done	Execution Arrow
Computer	Executes the code from the beginning	Manipulatives
Writer	Writes the code.	Pencils or Blocks
Driver	Proposes ideas for new lines of code; group then discusses.	All

# Resources

[Powerpoint:Lesson 1](#)

**Attention getting signals:**

- [One](#)
- [Two](#)
- [Three](#)
- [Four](#)
- [Five](#)

**Working in groups:**

- [Teaching Channel](#)

**Choral Response:**

- [Video](#)



## DO NOW



**Length:** 5 minutes

Introductions, students make name cards you can use for cold calling.

**Prep:**

- Index Cards
- Pen or Pencils

Teacher Actions	Student Actions
<p>1 Circulate room as students answer the do now on their paper. (3 min)</p>	<p>1 Students write their name on one side of their tent and on the other side they draw or write the answer to this question: When I grow up I want to be a(n) ____.</p>
<p>2 Whole group: (2 min)</p> <ul style="list-style-type: none"><li>• Introduce yourself</li><li>• Have one student at each table collect the name cards and hand them to you.</li></ul>	<p>2 Students place index cards at the top of desk.</p>



## ATTENTION GETTING SIGNAL



Length: 2 minutes

Teach class your attention getting signal.

Teacher Actions	Student Actions
<p>1 There are many times during this lesson where you will need to use an attention getting signal. Use a signal that you already have or grab some ideas from our Resources.</p>	<p>1 Students practice the attention getting signal.</p>



## CROSSING THE RIVER



**Length:** 30 minutes

**River Crossing Activity:** Students repeatedly encounter failure and connect it to progress as they solve the river crossing challenge.

**Prep:**

- Code Cards
- Felt Strips
- Paper River Crossing Worksheet
- Characters
- Classroom River
- Teacher Magnetic Code Cards & Step Arrow
- Teacher Characters

Teacher Actions	Student Actions
<p><b>1</b> Introduce What is Code: (1 minute)</p> <ul style="list-style-type: none"> <li>• Over this school year we are going to meet as a class once a week to build our skills as computer coders.</li> <li>• Can anyone tell me what is code?</li> <li>• As a _____ (profession a student wrote</li> </ul>	<p><b>1</b> Students raise their hands to share their ideas about what is code and what is a coder.</p> <ul style="list-style-type: none"> <li>• Answer: A set of instructions designed to be carried out by a computer. It is the instructions someone wrote to make your website, app, game, etc. run.</li> </ul>

on their card)  
how could you  
use code?

- Answer: Design your own website, build your own app, etc.

It is important that students don't say that using a website or computer is code. The distinction is that learning to code allows them to produce, not just consume.

2

### Introduce Productive Failure: (1 minute)

- Today we're going to become coders as we tackle a difficult challenge.
- Our code will not be carried out by a computer today, but it is going to mimic the process of writing code for a computer.
- We are probably not going to get it right the first time. It may take us getting it wrong a lot before we find our solution!

2

Students choral respond keywords on teacher's queue to increase engagement.

That is what happens when you write code.

- When coders get it wrong it is called a bug.
- Debugging is at the heart of the practice of coding. So today it is ok when your solution is wrong. Nothing bad is going to happen.

3

Reveal the Mission:  
(3 minutes) A farmer needs to cross a river with a chicken, a fox, and a bag of grain. However, his boat can only hold him and one other object. If left together, the fox will eat the chicken and the chicken will eat the grain. You need to get the farmer across the river without losing any of them.

3

Students follow along with mission on the board and answer CFUs.

- Answer: The fox will eat the chicken.
- Answer: The chicken will eat the grain.
- Answer: No, the fox and chicken only act up if the farmer crosses the river.

If a student has

<p>done this puzzle before it is not likely they have the solution memo rized. Ask the student to write ou t the solution an d if they have it that student can become your help er for the day.</p>	<ul style="list-style-type: none"> <li>• What happens if the chicken and the fox are left alone while the farmer goes in the boat?</li> <li>• If left alone, what will the chicken eat?</li> <li>• If the farmer is with them, will the chicken still eat the grain?</li> </ul>
<p><b>4</b> Demo a round of problem solving with roles as a whole class: (5 mins)</p> <p>Driver: (teacher) Ask students to propose a solution. Writer: Move the code cards to match the proposed solution. Stepper: Move the execution arrow and read each line of code one at a time. Computer: Move</p>	<p><b>4</b> Student volunteers act as: Writer: Move the code cards to match the proposed solution. Stepper: Move the execution arrow and read each line of code one at a time. Computer: Move the magnetic pictures on the board.</p>

the magnetic pictures on the board.

Suggested script when our solution is wrong:  
Our solution is wrong. We've got to start over. It's part of being a programmer!



5

Group Tackle: (15 minutes)

- Instruct students to draw their roles and connect them to their peer models in step 4
- Instruct students to pull materials from envelopes and work in their groups
- Pause every 3-5 minutes to troubleshoot as

5

Working in groups:

Students draw roles from the bags. Students follow their roles to assemble their code linearly on the felt. Groups that have the correct answer early: there are multiple solutions. Challenge them to find another solution.

a whole class  
and reveal the  
following hints:

1. Chicken  
has to  
cross the  
river first.
2. A  
character  
can cross  
the river  
more than  
once.
3. The farmer  
has to take  
either the  
chicken or  
the other  
item back  
across the  
river on his  
fourth trip.

6

Come to a solution:  
(5 min)

Call on a different  
group to share their  
solution. Follow the  
guidelines on whole  
group sharing.

- Help students  
come to the  
solution.
- Act out the  
solution as you  
step through  
each code card  
on the board.

6

One group brings  
solution to the  
board and perform  
it using their roles.

7

Direct students to return materials to the envelopes.

7

Students return materials to envelopes.



## NORM SETTING



**Length: 10 minutes**

Students reflect on how it feels to be wrong as they set norms for how they will treat themselves and their peers when they encounter failure.

**Prep:**

- Sticky Notes (2 colors, class set of each color)

Teacher Actions	Student Actions
<p>1 Whole class have students identify how it felt to get a solution wrong.</p> <ul style="list-style-type: none"> <li>• How does it feel when you are wrong?</li> <li>• As coders, we are going to be wrong. It is part of the process of creating good code.</li> <li>• It is about how we act and what we do when we fail that defines the kind of coder we are.</li> </ul>	<p>1 Students raise their hands to share their answers to the question.</p>

2

Answer questions on stickies:

- We are going to define how we can support each other when we have a bug.
- Students may need more prompting with "how does it look like/sound like when..." as follow up questions to oversimplified answers.

2

Individually students answer the two questions on their sticky notes.

1. How do you want to be treated when you're wrong?
2. How can you manage your fear when you're wrong?

3

By the end of this activity you should have a succinct set of norms for how the class wants to be treated when they fail.

- Collect stickies on board
- Group together similar ideas and read aloud.
- Students give thumbs up when they agree

3

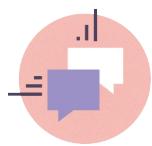
One student in each group collects the stickies and brings them up to the appropriate place on the board. Students respond with a silent thumbs up if they agree with a statement.

4

**Close out the lesson by acknowledging their great ideas and work they have accomplished.**

Suggested script:  
You have such great ideas! I am going to write these up so that everyone can see your work and bring it to our next class. This is going to help guide us this year as we become coders in how we are going to treat ourselves and each other when we encounter bugs in our code or an idea of ours fails. The lesson is that you have to fail before you succeed. Thank you so much for sharing your time and ideas with us today. I can't wait to tackle our next coding problem next week.

## Lesson 2: Getting Started on Scratch Plugged



## OVERVIEW

In this lesson, students will learn to login to Scratch and begin to use the editor, block palette, and stage to sequence a program.

# Print PDF

[Print this lesson](#)



## OBJECTIVES

---

- I can login, save my work, title my work using naming conventions, and organize my projects in Scratch.
- I can use the editor, block palette, and stage in Scratch to code my program.
- I can decompose a maze game into its fundamental requirements: hero, enemy, obstacles, goal.



## AGENDA

---

- Do Now (5 min) - Students see language written in block code and interpret its meaning, identifying that programming language is not the same as how we speak and write.
- Logging-in (15 min): Students login to their Scratch accounts.
- Saving Work (10 min): Students learn how to save and title their projects.
- Starter Maze (10 min): Students are introduced to Escape the Maze and practice reading and predicting how blocks of code will perform in the Scratch stage.
- Brainstorm (5 min): Students brainstorm ideas for building out the Escape the Maze game.
- OPTIONAL: Exit Ticket (15 min): Students predict how simple sequences of blocks will perform in the Scratch stage.

# Resources

[Powerpoint: Lesson 2](#)



## VOCAB

---

- **Editor:** A program designed for editing computer code by coders.



## MATERIALS

---

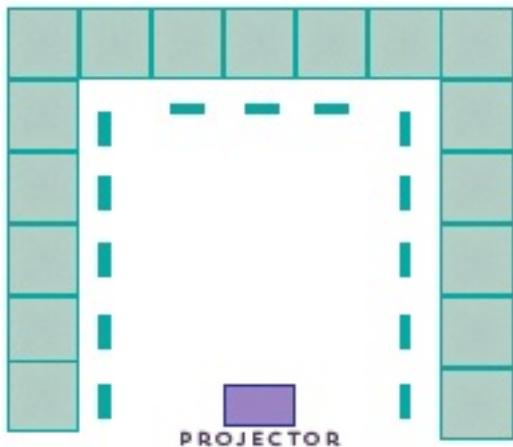
- Projector
- Do Now (class set)
- Computers (class set)
- Scratch Log-In Stickers
- Ideas Journals (class set)
- Exit Ticket (class set if printing, otherwise on Socrative)
- Chart Paper (1 piece)

## Notes

Prior to this lesson pass out the Scratch log-in stickers and have students place them on the inside cover of their idea journals.

This lesson includes several sets of steps that walk students through navigating the Scratch page. While the lesson includes suggestions for how to chunk them, you should modify the number of steps they are to follow at one time based on your students' familiarity with navigating websites and their ability to follow a sequence of instructions independently.

## Ideal Desk Setup



## Lesson Resources

If coding is new to your students we suggest showing them the Code.org short [Code Stars](#)

Pass out idea journals and have students place login stickers on the inside cover of their journals. Journals should be passed out and collected at the beginning and ending of each class.



## DO NOW



**Length:** 5 minutes

Students translate a block code version of “If you give a mouse a cookie” into a narrative and identify that computer code is different than the narrative language we use when telling a story or talking to each other.

**Prep:**

- Do Now worksheet

Teacher Actions	Student Actions
<p>1 Circulate room as students answer the do now on their paper.</p>	<p>1 Students independently answer the do now on their paper</p> <pre> when mouse receives cookie   say [May I have a glass of milk?]   say [May I have a straw?]   say [May I have a napkin?]   point towards mirror   think [Do I have a milk mustache?] </pre>
<p>2 After 3 minutes ask volunteers to read their interpretations. Read through the blocks so students can hear the</p>	<p>2 Students raise their hands to volunteer their interpretations of the sequence and answer the follow up questions.</p>

sequence flow.  
Follow up questions  
for the class:

- Does anyone recognize what book this is from?
- Is this how we talk or tell a story?
- When have you seen a sequence that looks like this before?
- Is the mouse ever going to ask for a straw before it asks for a cookie?

- Answer: If You Give a Mouse a Cookie
- Answer: No
- Answer: we used block code in last week's lesson to write the instructions for crossing the river
- Answer: No, because the code order determines the order of the story.

3

Connect the Do Now to the previous lesson in which students used paper code blocks to transport animals across a river.

**Key points:**

- Code is a set of instructions designed to be carried out by a computer.
- In the previous lesson we used paper code to transport animals across a river. We

3

Students are facing teacher, backs to computers, listening to the teacher's explanation.

acted as the computer in that lesson.

- In this Do Now the instructions are also in the form of blocks.
- Today we get to explore coding blocks on a computer.

Key points to hit:

- The language we use when we program a computer is not the same as the way we speak or write.
- Computers have their own languages and as we become coders we learn to write in the computer's language.



## LOGGING-IN AND EXPLORING



**Length:** 10 minutes

Students log-in and explore the Scratch studio.

**Prep:** Idea Journals and Log-in Stickers

Teacher Actions	Student Actions
<p><b>1</b> Introduce Scratch as the program we will be using to code.</p> <ul style="list-style-type: none"> <li>Has anyone used Scratch before?</li> </ul> <div style="background-color: #f0f0f0; padding: 10px;">           Acknowledge students who have. They will be a great asset to their peers as we all get comfortable navigating the program.         </div>	<p><b>1</b> Students are faced away from their computers, towards the teacher.</p>
<p><b>2</b> Show the [Getting Started with Scratch Video]. (<a href="https://scratch.mit.edu/help/videos/#">https://scratch.mit.edu/help/videos/#</a>)</p>	<p><b>2</b> Students watch video on projector.</p>
<p><b>3</b></p>	<p><b>3</b></p>

## Lesson 2: Getting Started on Scratch

<p>Point students to their log-in stickers inside their journals.</p> <ul style="list-style-type: none"><li>• Hold up the idea journal to model where they put their stickers.</li><li>• If students haven't put stickers in journals yet, do this now.</li></ul>	<p>Students hold up their idea journals and point to their stickers.</p>
<p>4 Remind students of the computer contracts they signed and expectations while we are on computers:</p> <ol style="list-style-type: none"><li>1. Raise your hand silently if you have a question.</li><li>2. When you complete a step give 2 silent thumbs up.</li><li>3. Close your laptops all the way when instructed.</li></ol>	<p>4 Students practice the 3 expectations.</p>
<p>5 Step through the 5 steps they will follow on their computer.</p>	<p>5 Students walk through the steps to log-in to their Scratch accounts and begin to</p>

## Lesson 2: Getting Started on Scratch

- If students are independently computer literate:
- Give the go signal for students to turn and face their computers.
- Circulate the room helping students who are stuck. If students are not independently computer literate:
- You may need to teach students how to type an underscore (\_)
- Go slide by slide through each step having students complete the slide with you

explore what they can do in the Scratch editor. Step 1: Go to scratch.mit.edu Step 2: Click the **Sign in** button in the upper right Step 3: Enter your username and password from your sticker and answer the questions when prompted. Step 4: Click on the "Exploring Scratch" studio and open "Make Something" Step 5: Create! Try making the sprite do something.

6

Show the "Make Your Sprite Move Forward" video and give students time to explore the Scratch editor.

- Practice dragging blocks in the script area, connect them, and click the

6

Students watch the video explore the Scratch editor.

green flag to run their program.

- If a student is more advanced and experienced with Scratch, point them to the challenges in the studio.

Hints in Scratch to share if students are struggling:

- If you don't know what a block of code does click the ? in the menu bar.
- Click on the top block to drag without separating blocks.
- Clicking the bottom block will detach that piece.

Attention getting signal

- What did you learn about Scratch?
- What did you observe in your exploration?



## SAVING WORK ON SCRATCH



**Length:** 10 minutes

Students learn to save and title their work. If there is time, share volunteers' programs.

Teacher Actions	Student Actions
<p><b>1</b> Step through the steps students will follow to save their work. If students are independently computer literate:</p> <ul style="list-style-type: none"> <li>• Give the go signal for students to turn and face their computers.</li> <li>• Circulate the room helping students who are stuck. If students are not independently computer literate:</li> <li>• Go slide by slide through each step having students complete the slide with you</li> </ul>	<p><b>1</b> Students are facing teacher, backs to computers, listening for instructions. Step <b>Remix</b></p> <p>1: Click in the top right of the editor. Step 2: Click on the title to rename your project. Step 3: Delete "remix" from the title and write "Challenge #_username" Step 4: Click "Share"</p>

<p><b>2</b> When 100% of the class has their programs named and remixed, if there is time:</p> <ul style="list-style-type: none"><li>• Who created something in Scratch and will let us view their code?</li></ul>	<p><b>2</b> Students raise their hands to volunteer to show their code.</p>
<p><b>3</b> Select a student volunteer from your class on Scratch and project their program on the big screen.</p> <ul style="list-style-type: none"><li>• This is an opportunity to reference the norms the class drafted in lesson 1 of how they want to be treated when they fail. Recognize the student's bravery for showing off their code.</li></ul>	<p><b>3</b> Students watch the screen at the front of the room and positively support their peers who demo their work.</p>

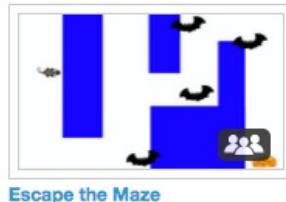


## STARTER MAZE



**Length: 15 minutes**

Students explore the maze project and predict how the code performs in the game

Teacher Actions	Student Actions
<p><b>1</b> Introduce the maze project!</p> <ul style="list-style-type: none"> <li>• Who here has done a maze before?</li> <li>• Who can tell me what is a maze?</li> </ul> <p>Dramatically reveal Escape the maze:</p> <ul style="list-style-type: none"> <li>• Part videogame, part maze</li> <li>• Students are going to learn to code like app developers!</li> </ul>	<p><b>1</b> Students raise their hands to answer question. Answers should include:</p> <ul style="list-style-type: none"> <li>• A path you need to navigate to the end</li> <li>• A start and an end/goal</li> <li>• Sometimes there are dead ends or obstacles</li> </ul> <div style="text-align: center;">  <p><b>Escape the Maze</b></p> </div> <p>Suggested script:</p>

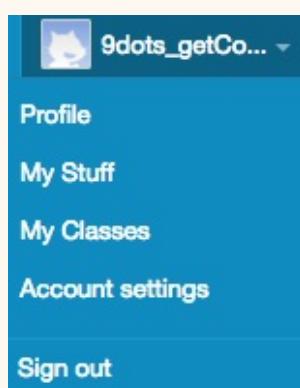
Now that you played around in the coding editor of Scratch, I'm going to let you in on the big surprise. This year, you're not just becoming coders, you're going to learn to code like a game developer! You're going to build your very own video game: Escape the Maze! I think you are ready to see some of the code right now.

2

Step through navigating to the starter maze and give them 2 minutes to play the game.

2

Step 1: Click on your username in the upper right and select "My Class"



Step 2: Click on the class studio "Escape the maze" Step 3: Open the starter maze, read the instructions, and click



to play the game.

<p><b>3</b></p> <p>Attention getting signal</p> <ul style="list-style-type: none"> <li>• What is happening in the game?</li> <li>• What happens if the ball touches the blue?</li> <li>• What does the blue represent?</li> <li>• What is the green block?</li> <li>• What happens when the ball gets to the green block?</li> </ul>	<p><b>3</b></p> <p>Students share out what happened in the game.</p> <ul style="list-style-type: none"> <li>• Answer: They are moving the ball through the maze.</li> <li>• Answer: The ball bounces backwards</li> <li>• Answer: The walls of the maze</li> <li>• Answer: The end, finish, goal</li> <li>• Answer: "You win!"</li> </ul>
<p><b>4</b></p> <p>Click to look at the code inside. Go through each block of code and ask students to connect it to the observations they made above about how the game is performing.</p> <ul style="list-style-type: none"> <li>• What is each sequence of blocks telling the program to do?</li> <li>• Give students 1 minute to discuss with their partners</li> </ul>	<p><b>4</b></p> <p>Students Think Pair Share to predict how the shown block of code is performing in the stage.</p>

## Lesson 2: Getting Started on Scratch

what the code  
is doing before  
sharing out.



## BRAINSTORM



**Length: 5 minutes**

Students brainstorm ideas for building out their maze in a list that we will reference as they develop their projects.

**Prep: Chart Paper**

Teacher Actions	Student Actions
<p><b>1</b> Brainstorm improving the game and record answers on chart paper or in a document:</p> <ul style="list-style-type: none"><li>• Think about some of the other games that you play. What could we add to Escape the Maze to make it more fun to play?<ul style="list-style-type: none"><li>◦ Call on 2 students to share.</li><li>◦ Give table partners 2 minutes to discuss and write ideas in their idea journals.</li></ul></li></ul>	<p><b>1</b> Students brainstorm ideas with their table partners and write in their idea journals.</p>

	<ul style="list-style-type: none"><li>○ If a student says they like a particular game (e.g. Pokemon) ask them what features they like about the game (e.g. you can earn points).</li></ul>
2	<p>Attention getting signal</p> <p>Call on students to share out their ideas:</p> <ul style="list-style-type: none"><li>● Continue to record their ideas</li><li>● It's totally okay to write down an idea if your friends come up with it.</li></ul>
3	<p>Close out this section of the lesson by letting students know you are excited about their ideas and the games they will code.</p>

## Lesson 2: Getting Started on Scratch

Suggested script:  
These are all great ideas. We will add them to our game next time. We will start to build onto the Escape the Maze game to make it more fun. As you become master coders, you are going to take over and design your own game.  
Keep brainstorming new ideas to put into your game.  
I can't wait to see the games you develop!



## EXIT TICKET



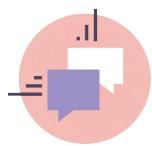
Length: 15 minutes

Prep: Socrative Quiz SOC-23584751

Teacher Actions	Student Actions
<p>1 Introduce the exit ticket:</p> <ul style="list-style-type: none"><li>• This is an opportunity for me to see where we stand as a class.</li><li>• It is important for you to try your best, but understand that some of these questions have been designed to be challenging and we are going to have time in upcoming lessons to follow up on questions you don't know yet.</li></ul>	<p>1 Students independently complete the exit ticket on Socrative.</p>
2	

Launch the exit ticket at [t.socrative.com](https://t.socrative.com). Walk students through what Socrative looks like and how to answer the questions on [m.socrative.com](https://m.socrative.com). You will need to give them the room name. SOC-23584751

## Lesson 3: Maze Scavenger Unplugged



## OVERVIEW

In this lesson, students learn the importance of giving clear and explicit directions. They practice sequencing code, writing code, and navigating their robot on a paper grid.

# Print PDF

[Print this lesson](#)



## AGENDA

---

1. **Warm Up:** Students are introduced to the importance of writing clear instructions by writing a set of instructions for their teacher to make a peanut butter and jelly sandwich. (10 min)
2. **Navigate the Maze:** The class is introduced to sequencing code by navigating their teacher to a spot on a grid. (10 min.)
3. **Maze Scavenger Hunt:** Students practice sequencing code in groups by writing code to collect prizes on a grid. (20 min.)
4. **Collect the Dots Extension:** Students continue practicing in groups by writing code to collect dots on a grid. (15 min.)
5. **Exit Ticket:** Students write code to navigate a robot to a spot on a paper grid. (5 min)



## VOCAB

---

- **Code:** A set of instructions designed to be carried out by a computer.
- **Validate:** To check if something is correct or does what it is intended to do.

# Resources

[Powerpoint: Lesson 3](#)



## MATERIALS

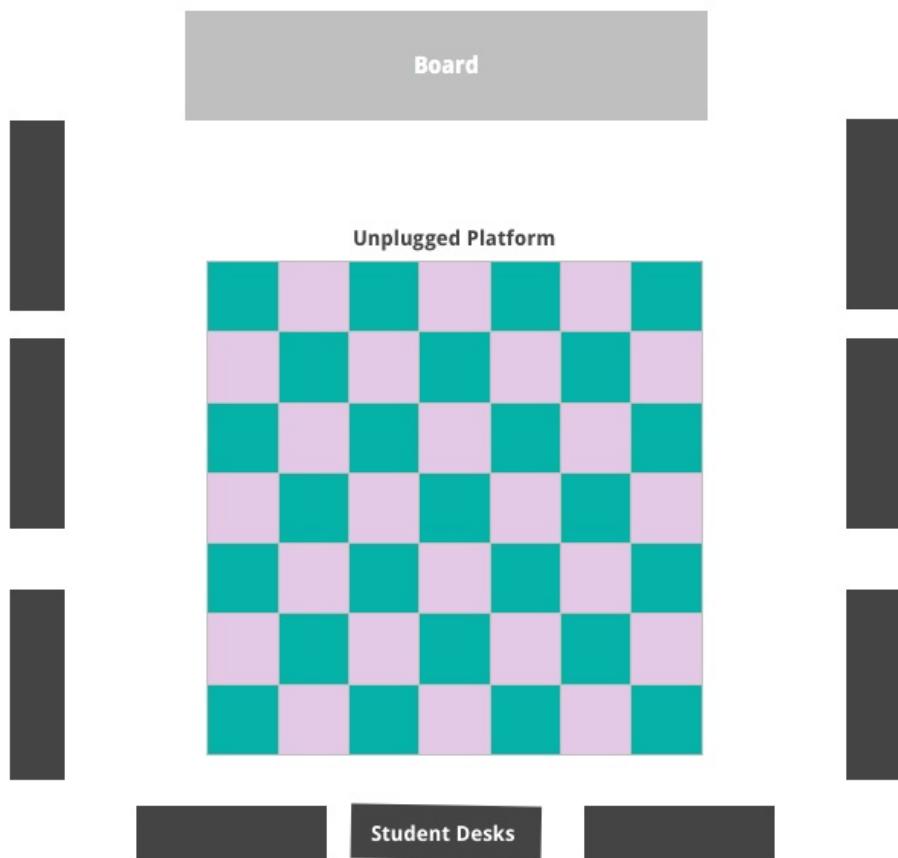
---

- Peanut butter
- Plastic butter knife
- Jelly
- Bread
- Paper plate
- [Warm up handout](#)
- [Paper maze worksheet](#)
- [Scavenger hunt maze worksheet](#)
- Role cards
- [Rover cutouts](#)
- Brown paper bag
- Large maze cutouts
- [Collect the Dots worksheet](#)
- [Exit ticket](#)
- Unplugged platform
- [Rover code blocks](#)

# Classroom Setup

## Lesson 3: Maze Scavenger Hunt

---





## WARM-UP



Length: 10 minutes

Students learn about the importance of giving clear and precise instructions by writing step-by-step instructions to make a peanut butter and jelly sandwich.

### Materials:

- Warm Up Handout
- Peanut Butter
- Jelly
- Bread
- Paper Plate
- Plastic Butter Knife

### Participation: Individual

Teacher Actions	Student Actions
<p>1 Have students respond to the prompt below in 5 minutes.</p> <ul style="list-style-type: none"><li>• Using the materials on the front table, write a set of step-by-step instructions for your teacher to make a peanut butter and jelly sandwich.</li></ul>	<p>1 Students write instructions for their teacher to make a peanut butter and jelly sandwich.</p>

**2** Ask for volunteers to share out their instructions in a step-by-step manner. Follow each step as the student reads it. If a student says to put the peanut butter on the bread, put the jar of peanut butter on top of the bag of bread. The point is to show the importance of giving clear and precise instructions. Allow at least two different students to share.

**2** Volunteers share their instructions with the class as the teacher attempts to follow along.

**3** Key Points

- Giving clear and precise instructions is important.
- Humans can often interpret the meaning and intentions behind a set of instructions in ways computers can't.

## Lesson 3: Maze Scavenger Hunt

Example: A person can say “you open door” to another person and that person would be able to determine that they are being instructed to open a door.

- In this class we will be writing clear and precise instructions that a computer can follow.



## NAVIGATE THE MAZE



**Length:** 10 minutes

Students are introduced to sequencing by coding instructions to navigate their teacher through a maze.

**Prep:**

- Paper maze worksheet
- Rover cutout
- Unplugged platform
- Rover code blocks

**Participation Whole Group and Individual**

Teacher Actions	Student Actions
<p><b>1</b> Introduce the class to Rover the robot. Just like other computers, Rover can only follow clear and precise instructions in the form of code. Code is a set of instructions designed to be carried out by a computer. Rover the robot can only follow three instructions.</p> <ul style="list-style-type: none"> <li>• move 1 step</li> <li>• turn left</li> <li>• turn right</li> </ul>	<p><b>1</b> Students listen to the introduction to Rover the robot.</p>

**2****Rover's Instructions**

Explain Rover's three instructions.

- move 1 step: makes Rover move one square in the direction it is facing (direction its headlights are pointing).
- turn left: makes Rover turn to its left. Have students point to their left. Then have them point to the direction Rover would turn if it was instructed to turn left. Explain that Rover's left is different than ours when it is facing in a different direction than we are.
- turn right: makes Rover turn to its right. Have students point to their right. Then have them point to the direction Rover would turn if it

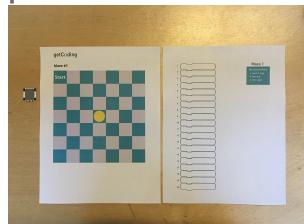
**2**

Students follow along by responding to teacher's prompts.

## Lesson 3: Maze Scavenger Hunt

<p>was instructed to turn right. Emphasize that our left and right is different than Rover's when it is facing in a different direction than we are.</p>	
<p><b>3</b> Check for Understanding</p> <p>Perform a check for understanding. Show the first Check for Understanding slide and ask the class to point to the direction Rover would turn if it was asked to turn left. Show the next slide and repeat the same prompt.</p>	<p><b>3</b> Students respond to the teacher prompts by pointing in the direction Rover would turn.</p>
<p><b>4</b> Have two student volunteers pass out robot cutouts and paper maze worksheets. Students should have their materials laid out in front of</p>	<p><b>4</b> Volunteers pass out materials and students lay out their materials in front of them.</p>

them like in the picture below.



5

Introduce the Maze Challenge

Explain the goal and rules of the first challenge.

- We will try to navigate Rover to the yellow spot on the grid by working together as a class.
- We can navigate Rover by using the only three instructions it can follow.
- We will sequence the code by writing the directions in the blank code blocks.

Describe the roles that students will play during the activity.

1. Driver:  
Proposes idea for new lines of

5

Students listen to the introduction of the maze challenge.

## Lesson 3: Maze Scavenger Hunt

- code; group discusses
2. Writer: Writes the code.
  3. Stepper: Keeps track of what the code has done
  4. Computer: Executes the code from the beginning.

Explain who will play each of the roles during the activity.

- The driver will be played by volunteers in the class.
- The writer will be a volunteer that will stick code blocks on the board.
- The stepper will be a volunteer that moves Rover on the unplugged platform.
- The computer will be played by the whole class when we read the code out loud.

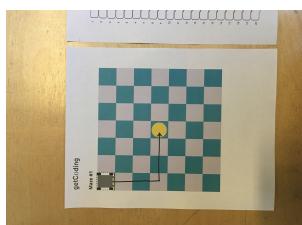
6

Ask the class for volunteers to play the role of the

6

Students volunteer to play the role of the stepper and

## Lesson 3: Maze Scavenger Hunt

writer and stepper.	writer during the maze challenge.
<p>7 Complete the Challenge:</p> <p>Have students place their Rover cutout at the starting square of their paper maze. Explain that the first step in completing this challenge is planning the path the Rover will take to get to the yellow spot. Show that there are many paths that can be taken by displaying three different choices in the slides.</p>	<p>7 Students place their Rover cutout on the starting spot on the grid.</p>
<p>8 Have the class vote on the path they will take and have them trace the path on their paper grid like in the picture below:</p> 	<p>8 Students vote on the path to take to get to the yellow spot on the grid and trace the path on their paper mazes.</p>
<p>9</p>	<p>9</p>

Ask the class what the first instruction should be.

Highlight that this is an example of the class playing the role of the driver.

Students share out possible instructions.

10

Once the class has settled on an instruction, have the writer place a code block on the board and have the stepper move Rover in response to the instruction.

10

The writer places a code block on the board and the stepper moves Rover on the unplugged platform.

11

Instruct students to write the instruction into the first blank code block on their worksheet and to move their Rover cutout on their paper maze. Continue this process each time an instruction is chosen by the class.

11

Students write the code into their code block and move Rover on their paper maze to match the position of Rover on the unplugged platform.

- Inform the class that you will be cold calling students for instructions after a volunteer shares the

## Lesson 3: Maze Scavenger Hunt

second instruction for Rover.	
<p><b>12</b> After the volunteer shares the second instruction for Rover, have the writer place the second code block on the board and have the stepper move Rover in response to the instruction.</p>	<p><b>12</b> A volunteer shares out the next line of code. The stepper updates the code on the board and the stepper moves Rover on the unplugged platform. All students update their code and move Rover on their paper mazes.</p>
<p><b>13</b> Cold call different students for the rest of the sequence.</p>	<p><b>13</b> Students called on share out the next line of code.</p>
<p><b>14</b> When the sequence is complete, have the stepper place Rover back at the starting square. Have the class read the code in unison and have the stepper move Rover as each block of code is read to check if the code is correct.</p>	<p><b>14</b> The stepper moves Rover back to the starting spot. The class reads the code in unison as the stepper moves Rover in response to the lines of code that are read.</p>

Highlight that th

## Lesson 3: Maze Scavenger Hunt

<p>is is an example of the class playing the role of the computer.</p>	
<p><b>15</b> Plan, Code, Validate Process</p> <p>Describe the process the class just used to plan, write, and check the code.</p> <ol style="list-style-type: none"><li>1. Plan: Explain that the class first planned how to complete the challenge by choosing a path and tracing it out on the paper grid.</li><li>2. Code: Explain that the class coded the path one step at a time.</li><li>3. Validate: Explain that the word validate means to check if something is correct or does what it is intended to do. In this process the class validated the code by reading the code starting at the beginning</li></ol>	<p><b>15</b> Students listen to the process they just went through together.</p>

<p>to see if it did what it was intended to do (get Rover to the yellow spot).</p>	
<p><b>16</b> <b>Maze 2</b></p> <p>Change the position of Rover and the yellow spot on the grid to match maze two. Explain that in this challenge students will work independently to get Rover to the yellow spot by writing their own code using the paper maze and Rover cutout.</p> <ul style="list-style-type: none"><li>• Display the plan, code, validate process on the board and remind students that they should use this process to complete the challenge.</li><li>• Inform students that they will have 5 minutes to complete the challenge and that you will call on</li></ul>	<p><b>16</b> Students listen to the instructions for the next maze challenge.</p>

## Lesson 3: Maze Scavenger Hunt

	volunteers to share their code.	
17	Allow students to begin planning and writing their code.	
18	After 5 minutes have passed, have volunteers share out their code and act it out as they read it.	18 Volunteers share out their code.



## MAZE SCAVENGER HUNT



Length: 20 minutes

Students practice sequencing instructions by navigating a classmate to collect a series of prizes in a maze.

Prep:

- 1 Scavenger hunt maze worksheet per group
- 1 Rover cutout per group
- Brown paper bag
- 1 set of Role cards per group
- Prizes
- Unplugged platform

Participation: Groups of 4

Teacher Actions	Student Actions
<p>1 Introduce the Scavenger Hunt</p> <p>Introduce the class to the scavenger hunt challenge.</p> <ol style="list-style-type: none"><li>1. Divide the class into groups of 4.<ul style="list-style-type: none"><li>• Pass out 1 scavenger hunt worksheet and Rover cutout per group.</li><li>• Each group will have 7 minutes to write code to</li></ul></li></ol>	<p>1 Students move into groups of</p>

## Lesson 3: Maze Scavenger Hunt

<p>get Rover to collect prizes on the unplugged platform.</p> <ul style="list-style-type: none"><li>• Each person in the group will play one of the roles described earlier in the lesson.</li></ul>	
<p>2 Have a student reach into the bag and pass out the role cards to the rest of his/her groupmates.</p>	<p>2 Students receive the role they will play in the group from a group mate.</p>
<p>3 Explain the roles each person will play in the group.</p> <ul style="list-style-type: none"><li>• Driver: proposes lines of code and asks if the group agrees</li><li>• Writer: writes the code on the worksheet</li><li>• Stepper: moves the Rover cutout on the paper maze and plays the role of Rover when the team presents its code</li></ul>	<p>3 Students listen to the roles they will play.</p>

## Lesson 3: Maze Scavenger Hunt

- Computer: checks another group's code for by walking through their code from the beginning and presents his/her team's code by reading it out loud as the stepper acts out Rover's movements.
- Emphasize that this is not a competition.
- Project the scavenger hunt maze on the board and explain that each group will have 1 minute to trace the path they will try to code to collect prizes on the platform.

4

Start the 1 minute timer and allow teams to plan and trace their path on the scavenger hunt paper maze.

- After 1 minute has passed check in with each group to

4

Groups plan and trace the path they will take to complete the challenge.

## Lesson 3: Maze Scavenger Hunt

<p>ensure that they have a path traced out.</p>	
<p><b>5</b> After 3 minutes have passed, have all of the groups put down their materials and have the computers rotate to the next group to walk through the code.</p> <ul style="list-style-type: none"><li>• After 1 minute has passed, have the computers go back to their original group and allow each group to work for 3 more minutes.</li></ul>	<p><b>5</b> The person playing the role of the computer rotates and checks the code of a neighboring group.</p>
<p><b>6</b> After 3 minutes have passed, have groups put down their materials and call on groups to present their code.</p>	<p><b>6</b> Groups present and act out their code on the unplugged platform.</p>



## EXTENSION ACTIVITY



**Length:** 15 minutes

The class practices sequencing by writing code to collect dots on a grid without barriers.

**Prep:**

- 1 Collect the Dots paper worksheet per group
- 1 Rover cutout per group
- Unplugged platform

**Participation:** Groups of 4

Teacher Actions	Student Actions
<p><b>1</b> Explain that in this activity students are collecting dots by navigating a classmate (Rover) around a grid that is free of obstacles. Inform the class that they will be working with the same teams, but they will be choosing new roles.</p>	<p><b>1</b> Students listen to directions.</p>
<p><b>2</b> Instruct students to place their role cards back in the paper bag. Have them re-select cards</p>	<p><b>2</b> Student place their role cards back in the brown paper bag and re-select roles</p>

## Lesson 3: Maze Scavenger Hunt

<p>from the bag in order to choose their new role.</p>	
<p><b>3</b> Present the new grid layout and explain that there are many ways to complete the task. Inform students that they will only have 5 minutes to plan and code their route.</p>	<p><b>3</b> Students listen to instructions.</p>
<p><b>4</b> Set the timer for 5 minutes and allow groups to begin.</p>	<p><b>4</b> Groups plan and code their routes.</p>
<p><b>5</b> After five minutes have passed, have groups present their code in the same manner that the scavenger hunt code was presented.</p>	<p><b>5</b> Groups present their code.</p>



## EXIT TICKET



**Length: 5 minutes**

The class completes an exit ticket.

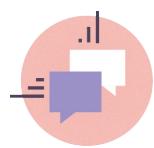
**Prep: Exit ticket**

**Participation: Individual**

Teacher Actions	Student Actions
<p>1 Say: Complete the exit ticket independently at your desk. This is your opportunity to show off what you know.</p> <p>Start 5 minute timer</p> <p>Collect exit tickets.</p>	<p>1 Students complete the exit ticket independently.</p>



## Lesson 4: Dance Off Online



## OVERVIEW

In this lesson, students learn how to plan, code, and validate their work by creating a dance off with their sprites on Scratch.

# Print PDF

[Print this lesson](#)



## OBJECTIVES

---

- I can code a sequence of actions in the order I want them performed.
- I can use the editor, block palette, and stage in Scratch to code my program.



## AGENDA

---

Do Now (5 min) - logging in

- Code Along (15 min): If You Give a Mouse a Cookie
  - Plan - read “If you give a mouse a cookie” instructions
  - Code - code along to the story
  - Debug - add wait blocks to debug
- Code Along (25 min): Dance Off *Plan - write out directions to your dance* Code - code your dance \*Debug - add wait blocks to debug
- EXTENSION (15 min): Coding Challenges



## VOCAB

---

- Editor: A program designed for editing computer code by coders.



## MATERIALS

---

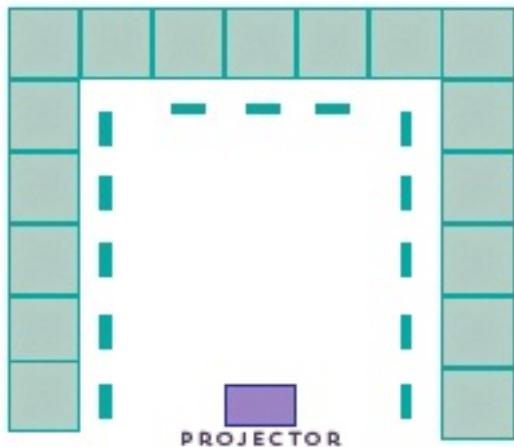
- Projector

## Lesson 4: Dance Off

---

- Idea Journals
- Computers (class set)
- Pencils (class set)

## Ideal Desk Setup



# Resources

[Powerpoint: Lesson 4](#)



## DO NOW



**Length: 5 minutes**

**Prep:**

- Computers
- Idea Journals
- Pencils

Teacher Actions	Student Actions
<p><b>1</b> Circulate room to assist students in logging-in to their Scratch accounts</p> <ul style="list-style-type: none"><li>• If necessary, review the computer usage expectations.</li></ul>	<p><b>1</b> Students log-in to their Scratch accounts.</p>



## CODE ALONG: IF YOU GIVE A MOUSE A COOKIE



**Length: 15 minutes**

Students code along with the teacher for 15 minutes. Stop the activity at 15 minutes to ensure time for individual coding with the Dance Off activity.

**Prep:**

- Computers
- Plan Chart
- "If You Give a Mouse a Cookie" book or audiobook Use [this Scratch Project](#) for student code along.

Teacher Actions	Student Actions
<p><b>1</b> Read "If You Give a Mouse a Cookie" to students (3 min)</p>	<p><b>1</b> Students sit facing teacher, computers closed.</p>
<p><b>2</b> Show the step by step plan (2 min) *Ask students to identify connection between a step and what happened in the story.</p> <p><b>Mouse Will:</b></p> <ol style="list-style-type: none"> <li>1. Ask for a glass of milk</li> <li>2. Go to Milk</li> </ol>	<p><b>2</b> Students identify that:</p> <ul style="list-style-type: none"> <li>• Steps 1 &amp; 2 are "... it's going to want a glass of milk."</li> <li>• Steps 2 &amp; 3 are "... it's going to ask for a straw."</li> </ul>

3. Ask for a straw
4. Go to straw
5. Ask for a napkin
6. Go to napkin
7. Look in the mirror
8. Go to mirror
9. Ask for a pair of scissors
10. Go to scissors

Code Along (10 min)

- Setup
  - Navigate to "My Classes", "Exploring Scratch" studio, and open project "If You Give a Mouse a Cookie"
  - Click the green flag to show that the project doesn't do anything yet... we need to code it!
  - Click "See Inside"
  - Click "Remix"
  - Re-title your work
- Coding

- Steps 3 & 4 are "...it's going to ask for a napkin."
- Steps 5 & 6 are "...it's going to ask for a straw." Continue until you feel students understand the sequence.

## Lesson 4: Dance Off

- The events block indicates when the sequence should begin:

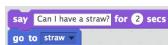


- Code the first 2 steps of the plan:



- Pause to run the program and check it is making sense

- What do you think the next two blocks will be?



- Continue to code and pause to check after every 1-2 lines you add. Check off the plan as you go through it. Every time you re-run it you will need to move the mouse

<p>back to its starting point.</p>	
<p><b>3</b> Check student work.</p>	<p><b>3</b> Students turn to their computers and follow the steps in the code along, giving thumbs up when ready for a next step.</p> <ul style="list-style-type: none"> <li>• Remind students to:       <ul style="list-style-type: none"> <li>○ Code the correct sprite</li> <li>○ Choose the “say for 2 sec” block</li> <li>○ Move the mouse back to its starting point before re-running the code</li> </ul> </li> </ul> <p><b>Finished program:</b></p>  <pre> when green flag clicked   say [I want milk] for [2] secs   go to [milk v]   say [Can I have a straw?] for [2] secs   go to [straw v]   say [May I have a napkin?] for [2] secs   go to [napkins v]   say [Hmm, do I have a milk mustache?] for [2] secs   go to [mirror v]   say [Ohhh I need to trim my hair!] for [2] secs   go to [broom v]   say [I need a broom to clean up the mess] for [2] secs </pre>



## CODE ALONG: CHOREOGRAPH A DANCE!



**Length: 25 minutes**

Students code along with the teacher until they are ready to plan, code, and validate their dance off code.

**Prep:**

- Idea Journals
- Computers
- Pencils

Teacher Actions	Student Actions
<p><b>1</b> Plan 3 step dance (5 min)</p> <ul style="list-style-type: none"> <li>• Introduce activity: "We are going to choreograph a dance for our Sprite."</li> <li>• Pick 5 dance moves for our sprite to execute (prioritize using the highlighted blocks):           <ul style="list-style-type: none"> <li>◦ Say</li> <li>◦ Move</li> <li>◦ Turn</li> <li>◦ Think</li> <li>◦ Change size</li> </ul> </li> </ul>	<p><b>1</b> Students volunteer 5 dance moves from the list to choreograph their dance.</p>

## Lesson 4: Dance Off

<ul style="list-style-type: none"><li>◦ Change color</li><li>◦ Next costume</li><li>• Write out the 5 dance moves in order on your planning chart</li></ul>	
<p><b>2</b> Code Along (10 min)</p> <ul style="list-style-type: none"><li>• Setup<ul style="list-style-type: none"><li>◦ Navigate to “My Classes”, “Exploring Scratch” studio, and open project “Dance Off”</li><li>◦ Click the green flag to show that the project doesn’t do anything yet... we need to code it!</li><li>◦ Click “See Inside”</li><li>◦ Click “Remix”</li><li>◦ Re-title your work</li></ul></li></ul> <p>Code</p>	<p><b>2</b> Students follow along on their computers as they code the dance for the first sprite.</p>

## Lesson 4: Dance Off

- We will begin when the green flagged is clicked:

when  clicked

- Drag coding blocks to represent the 3 actions and test code by pressing the green flag

### Adjust

- Play with what happens when you change numbers in each block. After each adjustment run the code again to see how it affected the dance.

### Repeat

- Copy and paste the blocks using the stamp tool so that it repeats the actions multiple times.

3

Students plan and code dances for Khalid (10 min)

3

Students plan in their idea journals their dance.

## Lesson 4: Dance Off

<ul style="list-style-type: none"><li>• (2 min) In idea journals have students write out their 5 step dances using the given blocks</li><li>• (8 min) Students delete the code we have and create their own</li></ul>	
<p><b>4</b> Share dances (5 min)</p> <ul style="list-style-type: none"><li>• Click “Share”</li><li>• Click “Studios” under your project</li><li>• Click the check mark next to “Dance Off”</li></ul>	<p><b>4</b> Students put their projects into the shared studio</p>
<p><b>5</b> Go over norms for viewing peer’s work</p> <ul style="list-style-type: none"><li>• Constructive Feedback: “It would be cool if...”</li><li>• Positive Speak: Tell someone what you like about their program before giving any constructive feedback</li></ul>	<p><b>5</b> Students read norms and share additional norms they would like their peers to adhere to.</p>

## Lesson 4: Dance Off

<ul style="list-style-type: none"><li>• No negative comments</li></ul>	
<p><b>6</b></p> <p>Students view their peer's work</p> <ul style="list-style-type: none"><li>• Click "Dance Off" to view everyone's dances</li></ul>	<p><b>6</b></p> <p>Students browse each other's projects in the studio.</p>



## EXTENSION ACTIVITY



Length: 15 minutes

If you have an 60 minute block for class, try this extension activity.

Prep:

- Computers

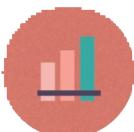
Teacher Actions	Student Actions
<p>1 Students can continue to improve upon their dances</p>	<p>1 Lab time</p>
<p>2 Or students navigate back to the “Explore Scratch” studio and attempt to solve the 3 challenge projects</p>	

## Lesson 5 In the Loop



### Overview

In this lesson students explore and predict how they can use loops to more efficiently write code. They will begin to transfer their coding concepts from a paper world to the unplugged platform and in the end will have code they are ready to test online in the next lesson.



### Objectives

- I can use the debugging process to debug my code.
- I can replace a repeating sequence with a loop to increase code efficiency and readability.



## Agenda

1. Do Now (10 min)
2. Engage: Sequence Through a Maze (5 min)
3. Explore & Explain: Introducing Loops (7 min)
4. Elaborate: Loops in the Real World (5 min)
5. Extension: Coding the Enemy (10 min)

Independent Coding Practice:

<https://studio.code.org/s/course2/stage/3/puzzle/1>



## Materials

Teacher Materials:

[Lesson 5 Slides](#)

Projector

Unplugged Foam  
Maze

Whiteboard

Teacher Scratch  
Blocks magnetic

Student Materials:

[Maze Handout](#)  
(class set) - Editable  
version here

Scratch block strips  
(1 set per group)

Felt (1 per group)

Role Cards (1 set  
per group)

Dry erase markers  
(class set)

Rovers (class set)



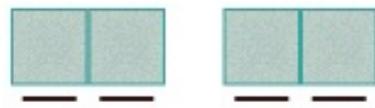
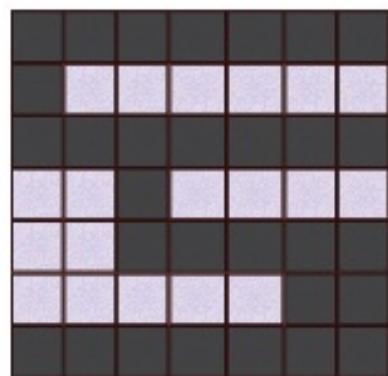
## Vocabulary

- **Loop:** A sequence of instructions that is continually repeated until a certain condition is reached.
- **For-Loop:** A type of loop that specifies the number of times to repeat the nested sequence of instructions.

# Room Design

**Group 1: Position desks around floor tiles.**

Teacher, Board, Projector



**Group 2: Ensure student screens are visible while you are teaching group 1.**



## Symbols Key

- help
- question
- check\_circle
- answer
- action item

## Do Now (10 min)

Explain the new format of our lessons:

- The class will be in 2 groups assigned by the teacher.
- While one group completes a coding lesson with the teacher, the other group practices a coding concept independently.
- Halfway through class the groups will switch so that everyone completes both sets of activities.

lightbulb\_outline

tip

*For younger students and students not used to moving around the classroom, have them practice the transition between groups while practicing moving quickly and safely to their new seat. Use a timer to encourage speed.*

Demonstrate the independent work

- Show students code.org using the tinyurl for the assigned lesson.
- Review expectations for using the computers independently. It is important students work silently and troubleshoot independently so that the group in the front of the room can hear the lesson.

Move students into groups

## Engage: Sequence Through a Maze (5 min)

**Practice Sequence Debugging:** Instruct students to move their rover around their paper maze to identify where in the given sequence there is a bug. Students can use a dry erase marker to tick off each line of code as it is completed.

**Do Now - Maze**

The maze is a 6x6 grid. The 'Start' is at the top-left corner (1,1). The 'Finish' is at the bottom-right corner (6,1). Blue vertical walls are located at columns 2, 4, and 6. Blue horizontal walls are located at rows 2, 4, and 5. The path from Start to Finish is: (1,1) → (1,2) → (2,2) → (2,4) → (3,4) → (3,6) → (4,6) → (4,5) → (5,5) → (5,2) → (6,2) → (6,1).

Start

Finish

1. move 6 steps
2. turn ↘ 90 degrees
3. move 2 steps
4. turn ↗ 90 degrees
5. move 2 steps
6. turn ↗ 90 degrees
7. move 2 steps
8. turn ↘ 90 degrees
9. move 3 steps
10. turn ↗ 90 degrees
11. move 6 steps

history

reminders

- Students may need to be reminded what a sequence is: the order in which the code is given.
- Students may also need a reminder of what 90 degrees clockwise and counterclockwise means.

### Students Propose Solutions:

help

If the code works as it is, stand up. If the code does not work, put your hands on your head.

check\_circle

Students stand up/sit down to respond. No, the code does not work.

help

How would you change the sequence of the code to get your hero to the finish line?

check\_circle

Answer: Students should identify that after line 10 there needs to be another move 2 steps and then turn right 90 degrees.

vpn\_key

### Key Points

- Validating your code is an essential step in writing code.
- When you test a sequence, point to

each block and act  
it out on the stage  
in order, one line at  
a time.

**Perform Solution:** Assign students to group roles to act out the code.



What did the code do before that wasn't working?



Answer: At line 11 the hero moved 6 steps instead of 2 so it walked off the board.

## Explore & Explain: Introducing Loops (7 min)

**Assign Group Roles:** Students are working in groups of 4 for this lesson and should pull group roles. Review each role's responsibilities.

**Discover Loops:** Working in groups

- **Plan:** Students use dry erase marker to draw the path the rover will take around their maze.
- **Code:** Students put together a sequence of code on their felt to circumnavigate the grid. Do not give them enough move and turn blocks. Each group will only have 2 of each type of block.



- **Validate:** Run your code from the beginning to test it.

lightbulb\_outline

Tip

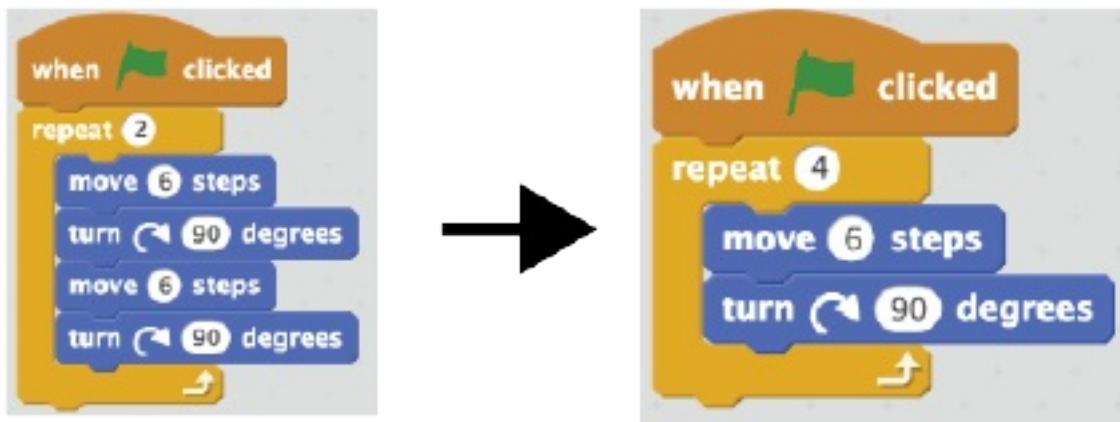
*Younger students may  
need to be told to write  
in 90 degrees on turn  
blocks.*

**Add Loop Blocks:** Wait until students identify that there are not enough blocks before introducing loops. Without introducing them verbally, place loop strips on each group's table:

**Demo Code:** Call on one group to demo their code by bringing their felt up to the board and acting it out on the unplugged maze.

help

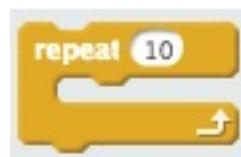
Extension: Can you rewrite your code with fewer blocks



Predict Vocabulary Definitions: Each group's driver holds up each loop as you introduce it.

help

For each loop: What do you think this loop will do?



check\_circle

Forever: Repeats the nested sequence forever

check\_circle

Repeat Until \_: Repeats the nested sequence until an action happens

check\_circle

Repeat \_\_: Repeats the nested sequence that number of times

vpn\_key

### Key Points

- These structures are called loops.
- Loops make it so we don't have to write a sequence of code over and over to make it repeat.
- In other programming languages we call

these for loops,  
forever loops, and  
while loops.

## Elaborate: Loops in the Real World (5 min)

**Teach the Loop Gesture:** Have students move their hand in the loop gesture to remember to read each nested line of code and then return to the start of the loop.

**Examples of loops in real life:** Have students identify the type of loop you would use for each of the examples below.

help

The School Week

check\_circle

Repeat Until Summer Break

help

The Seasons

check\_circle

Forever

help

Washing Dishes

check\_circle

Repeat Until No More Dirty Dishes

help

Riding The Tower of Terror 5 Times

check\_circle

Repeat 5

help

Riding The Tower of Terror Until the Park Closes

check\_circle

Repeat Until Park Closes

**Think Pair Share:** What other examples of loops can you come up with?

