

Ozobot Evo Coding Unit

Total Number of Lessons: 5
Total Estimated Time for Module: 450 min

SECTION 1: MODULE OVERVIEW AND CURRICULAR CONNECTIONS – REQUIRED

- **Module goal:** By the end of this module students will be able to use color coded lines and simple block programming to program their robots complete simple tasks
- **Connection to standards:** 3-5.PA.1, 3-5.PA.2, 3-5.PA.3, 3-5.PA.4, 3-5.PA.5, 4.G.4, 4.G.5, 4.M.1, 4.M.5, 4.M.6, 4.AT.6 (The module hits these Indiana computer science and math standards)
- **RET materials/ideas to be leveraged:** I will use the knowledge I gained on block programming to teach the students how to make simple programs for Ozobots. The Ozobots will be purchased through the program

SECTION 2: OVERVIEW OF MODULE FRAMEWORK – REQUIRED

- **Real-world context:**
 - **Traffic and road design** — Speed codes (slow near a "school zone," turbo on a "highway"), stoplights (pause codes), roundabouts (tornado/zigzag). Kids map their own neighborhood or a fictional town, which doubles as a geometry/measurement task.
 - **Delivery and logistics** — "Ozobot delivers packages" across a map with multiple stops, time limits (timer codes), and shortest-path decisions. Ties naturally into a discussion of how companies like UPS or Amazon route trucks.
 - **Search and rescue robot** — Ozobot has to "find" a person or object on a grid, reinforcing algorithmic thinking (the sequence of steps needed) and conditionals ("if it detects [color], it has found the target").
 - **Math word problems come alive** — Perimeter/area of the course they draw, angle measurement for turns, elapsed time for a "race," which lets Lesson 1–2 double as a math center.
- **Background STEM content:**

Light & Color (Physical Science)

This is the biggest one, and it directly explains *why* Color Codes work at all: The robot works by detecting how much light bounces back off a surface — black absorbs almost all light (low reflection), while red, green, and blue each reflect a distinct wavelength pattern the sensor can distinguish. This is a great tie-in to a basic light unit: light travels, hits a surface, and is either absorbed or reflected — and the robot is literally using that reflected light as data.

Electricity, Circuits & Energy

Evo has 6 programmable LEDs, and the LEDs display colors based on the line color it's following or can be set directly by the program — a concrete example of electrical energy converting to light energy. Evo uses wheels, a motor, and a battery to move, with the motor and battery connected to the wheels even though they aren't visible from outside — a simple, visible example of stored (battery) energy becoming motion (kinetic) energy, which connects to elementary energy-transformation content. Evo can be programmed to play sounds through a speaker located between the proximity sensors — ties to a sound/vibration unit (speakers vibrate to create sound waves).

- **Description of Main Activity/Project:** [Describe in 1-3 paragraphs]
 - Students will start working with the Evo Ozobots at a very basic level in the beginning lessons. They will program them using lines of various colors and patterns to get their Evo to perform different tasks. As the lessons progress they will have the opportunity to begin to use simple block programming to make their robots move and perform tasks. There are many connections that can be made between this module and math and science concepts required of 4th graders in Indiana. Some connections are light and sound and electricity and energy.

SECTION 3: MODULE SEQUENCING AND ASSESSMENT – REQUIRED
--

- **Brief description of sequenced learning objectives:** The students will start out by learning about the robot and sensors. Then they will use preset paths to move their bots. As the lessons progress they will be able to use the knowledge they have learned to make their own lines and use their own color codes to control the movement of their bots. Next they will learn about simple block programming and use it to program their robot for simple tasks. Finally they will bring both the lines and block coding together to complete a culminating project that tells a story or completes an obstacle course with their robot.
- **Brief description of formative and summative assessment approaches:** Students will be given opportunities to show that they are able to complete the tasks as we progress through the module. At the end of the module they will have the opportunity to complete a summative assessment using both line and block programming to design their own story or obstacle course.

- **Recommendations for implementation:** Take the time to experiment with the Evo robots and the block programming, so you are familiar with it before introducing it to your students. Have the students work on the floor so their robots don't fall off of a raised surface and break. Take the time to establish norms for working in small groups before handing out the robots and materials.

SECTION 4: MODULE FILES – ENCOURAGED/OPTIONAL

Below is copy of the lessons and the corresponding Indiana academic standards:

OZOBOT CODING UNIT

Five Lessons in Block Coding & Color Codes

Designed for Grade 4 · 45–90 minute sessions

Unit Overview

This unit introduces fourth graders to two complementary ways of programming an Ozobot robot: drawing Color Codes with markers, and building programs with block-based coding in OzoBlockly (ozoblockly.com). Lessons progress from simple sequencing to loops and conditionals, culminating in a project that combines both coding methods. This unit is designed to be used a lesson at a time throughout the school year to introduce students to Ozobots and reinforce computer science and math standards.

#	Lesson	Coding Focus	Duration
1	Meet Ozobot: Line Following & Color Code Basics	Color Codes (marker)	45–50 min
2	Ozobot City: Advanced Color Codes	Color Codes (marker)	45–50 min
3	Intro to Block Coding with OzoBlockly	Block Coding	45–50 min
4	Loops, Conditionals & Sensors	Block Coding	45–50 min
5	Culminating Project: Story or Race	Both combined	60–90 min

Core Materials Needed Across the Unit

- 1 Ozobot Bit or Evo per pair of students (fully charged)
- Ozobot Color Code markers: black, red, green, blue (fine-tip, included with Ozobot)
- White paper or cardstock, unlined, plus rulers
- 1 Chromebook, tablet, or laptop per pair with a web browser (for ozoblockly.com)
- Printed Ozobot Color Code reference chart (free download at files.ozobot.com; the most-used codes are reproduced in the Appendix of this packet)
- Charging cables and a power strip
- Student planning/storyboard sheets (one per lesson is suggested; teachers may create their own or use notebook paper)

LESSON 1

Meet Ozobot: Line Following & Color Code Basics

Duration	45–50 minutes
Coding Mode	Color Codes (hand-drawn, marker-based)

Learning Objectives

- Students will explain that Ozobot uses optical sensors on its underside to read lines and colors.
- Students will identify what a Color Code is and use three basic codes (Line Switch Left, Line Switch Right, U-Turn) to control Ozobot's path.
- Students will draw a legal Ozobot line and Color Code following spacing rules (line width, safe zones).

Standards Alignment

Supports CSTA Algorithms & Programming standards for grades 3–5 (creating and following step-by-step sequences/algorithms) and ISTE Standard 5, Computational Thinker.

Vocabulary

sensor · algorithm · sequence · color code · intersection

Materials

- 1 Ozobot (Bit or Evo) per pair, charged
- Black, red, green, and blue Ozobot markers
- White paper, rulers, pencils
- Printed Color Code reference chart (Appendix A)

Warm-Up (5 minutes)

Ask the class: "This little robot has no screen, no buttons, and no remote control — so how could it possibly know where to go?" Take a few guesses, then reveal that Ozobot "sees" using light sensors on its underside that scan the color of the surface beneath it many times per second.

Direct Instruction (10 minutes)

1. Place Ozobot on a pre-drawn straight black line and let it drive so students can watch the sensors in action.
2. Explain that a Color Code is a short pattern of colored stripes that gives Ozobot a specific instruction, the same way a word is a pattern of letters that gives a reader meaning.
3. Introduce the reference chart and demonstrate three codes live: Line Switch Left, Line Switch Right, and U-Turn.
4. Point out the drawing rules: lines should be about the width of the marker tip, and each code needs a small blank "safe zone" of plain black line before and after it so Ozobot can read it cleanly.

Guided Practice (10 minutes)

In pairs, students test a teacher-prepared worksheet with one of each basic code already drawn. Students predict what Ozobot will do at each code, then run it to check their prediction.

Main Activity (15 minutes)

1. Pairs sketch their own simple path in pencil first (a loop, a zigzag hallway, or a letter shape).
2. Students trace the path in black marker, then add at least one U-Turn code and one Line Switch code, leaving proper safe zones.
3. Pairs test their path with Ozobot, observe what happens, and adjust the drawing if a code doesn't trigger (usually a spacing or width problem).

Assessment / Check for Understanding

Exit ticket: students draw one Color Code from memory on an index card, label it, and write one sentence describing what it tells Ozobot to do.

Differentiation

- Support: provide a pre-drawn track with only one code to modify, rather than a full design from scratch.
- Extension: challenge early finishers to look up and add a Zigzag or Tornado code from the reference chart.

LESSON 2

Ozobot City: Advanced Color Codes

Duration	45–50 minutes
Coding Mode	Color Codes (hand-drawn, marker-based)

Learning Objectives

- Students will use speed codes (Slow, Fast, Turbo) and special-move codes (Zigzag, Tornado, Pause) purposefully in a design.
- Students will collaborate to plan, build, and debug a multi-code course that models a real-world system.

Standards Alignment

Supports CSTA Algorithms & Programming standards for grades 3–5 (sequencing multiple commands, testing and debugging a program) and ISTE Standard 5, Computational Thinker.

Vocabulary

debug · sequence · speed code · special-move code

Materials

- Same as Lesson 1, plus large butcher paper or poster board per group (roughly 2' x 3')
- Color Code reference chart (Appendix A)

Warm-Up (5 minutes)

Quick recall game: hold up code cards from Lesson 1 (Line Switch Left/Right, U-Turn) and have students call out what Ozobot will do for each.

Direct Instruction (10 minutes)

1. Introduce speed codes (Slow, Fast, Turbo) and connect them to real speed limits — school zones are slow, highways are fast.
2. Introduce Pause, Zigzag, and Tornado, and discuss real-world equivalents (a stoplight, a car swerving, a roundabout).

3. Model combining two codes in a single short stretch of road and predict the combined behavior before testing.

Main Activity: Design an Ozobot City (25 minutes)

1. In groups of 3–4, students plan a small city map on scratch paper: roads (lines), at least one intersection, and locations such as a school, a highway, and a stoplight.
2. Groups draw the final map on poster paper, including at least five different Color Codes that match the purpose of each location (e.g., Slow near the school, Turbo on the highway, Pause at the stoplight).
3. Groups test their city with Ozobot, debug any codes that don't trigger correctly, and refine their map.

Assessment / Check for Understanding

Gallery walk: each group briefly explains their map to a visiting group, pointing out where each code is and why they chose it there. Teacher circulates with a simple checklist noting whether each group can name at least three codes correctly.

Differentiation

- Support: provide sentence starters for the gallery walk explanation ("We used the ____ code here because ____.")
- Extension: challenge groups to add a Timer code so part of the city "closes" after 30 seconds.

LESSON 3

Intro to Block Coding with OzoBlockly

Duration	45–50 minutes
Coding Mode	Block Coding (ozoblockly.com)

Learning Objectives

- Students will describe the difference between a hand-drawn Color Code and a digital block program.
- Students will write a sequence of movement, light, and sound blocks in OzoBlockly and run it on a physical Ozobot.

Standards Alignment

Supports CSTA Algorithms & Programming standards for grades 3–5 (creating programs with sequences of instructions) and ISTE Standard 5, Computational Thinker.

Vocabulary

block · program · sequence · calibrate · upload

Materials

- 1 Chromebook/tablet/laptop per pair, browser open to ozoblockly.com
- 1 Ozobot (Bit or Evo) per pair, fully charged
- Projector for whole-class demonstration

Warm-Up (5 minutes)

Ask students how the color-code drawings from Lessons 1–2 are similar to and different from typing a set of instructions on a screen. Land on the idea that both give Ozobot a sequence of steps, but a block program can also add things a marker can't, like specific sounds and precise light colors.

Direct Instruction (15 minutes)

1. Open OzoBlockly together on the projector and select the beginner block level. Point out the categories of blocks: Movement, Light, Sound, and Loops/Control.
2. Drag a few movement blocks together (move forward, turn) to show that blocks snap into a single connected program, top to bottom, just like a written recipe of steps.
3. Demonstrate running the program: place Ozobot on the flashing calibration square on the screen so it can "download" the program through its light sensors, then set it down to watch it run.
4. Build a short class program together: move forward, turn, change LED color, play a tone.

Main Activity (20 minutes)

1. In pairs, students open a new OzoBlockly program and build a sequence that drives Ozobot in a square path (four move-forward blocks and four turn blocks).
2. Students add a Light block that changes the LED color at each corner of the square.
3. Pairs calibrate and test their program on a clear tabletop, then revise if Ozobot doesn't complete the shape as planned.

Assessment / Check for Understanding

Quick checklist: program includes at least four movement blocks and at least one light block; Ozobot successfully completes a recognizable square when tested.

Differentiation

- Support: give a partially built program (missing 1–2 blocks) for students to complete rather than building from a blank canvas.
- Extension: challenge students to add a Sound block that plays at the finish, or to shrink their program using a loop block (previewing Lesson 4).

LESSON 4

Loops, Conditionals & Sensors

Duration	45–50 minutes
Coding Mode	Block Coding (ozoblockly.com)

Learning Objectives

- Students will use a repeat (loop) block to shorten a repetitive program.
- Students will use an if/then (conditional) block with Ozobot's color sensor to make the robot respond to what it detects.

Standards Alignment

Supports CSTA Algorithms & Programming standards for grades 3–5 (using loops and conditionals, decomposing a problem into smaller parts) and ISTE Standard 5, Computational Thinker.

Vocabulary

loop · repeat · condition · if/then · decompose

Materials

- Same as Lesson 3
- Small squares of red, green, and blue paper or card (for testing the color sensor)

Warm-Up (5 minutes)

Unplugged loop game: call out a four-count action ("clap, turn, clap, turn") and have students physically repeat it four times. Ask: was it easier to give one instruction and say "repeat four times," or to say the whole thing four separate times? Connect this directly to the square program from Lesson 3.

Direct Instruction (12 minutes)

1. Open yesterday's square program and show how a Repeat block can replace the four repeated move-and-turn pairs with a single repeat-4 loop containing just one move and one turn block.
2. Introduce the if/then block paired with the color sensor block: "if Ozobot's sensor detects red, then flash lights and play a sound."
3. Model testing the conditional by holding a red paper square under Ozobot's sensor and watching the response.

Main Activity (25 minutes)

1. Pairs rebuild their square program from Lesson 3 using a Repeat loop instead of four separate move/turn pairs, and confirm it still draws a square.
2. Pairs add an if/then block that makes Ozobot react in a specific way when it detects one chosen color (e.g., flash red lights and play a low tone if it senses red).
3. Pairs test and debug both parts of the program, recording in their planning sheet what they changed and why.

Assessment / Check for Understanding

Rubric check: program correctly uses a loop to shorten repeated steps, and a conditional block produces the intended response to a detected color. Students briefly explain, in their own words, what "if/then" means in their program.

Differentiation

- Support: provide a program with the loop already built, so students focus only on adding the conditional.
- Extension: challenge students to nest a small loop inside another, or to add a second condition for a different color.

LESSON 5

Culminating Project: Ozobot Story or Race

Duration	60–90 minutes (may span two class periods)
Coding Mode	Both Color Codes and Block Coding, combined

Learning Objectives

- Students will design an original course or story that requires both a hand-drawn Color Code path and an OzoBlockly program.
- Students will apply sequencing, at least one loop or conditional, and at least three Color Codes purposefully.
- Students will present and explain their design choices to classmates.

Standards Alignment

Synthesizes CSTA Algorithms & Programming standards for grades 3–5 (sequences, loops, conditionals, testing/debugging) and ISTE Standard 4, Innovative Designer, and Standard 5, Computational Thinker.

Materials

- All materials from Lessons 1–4
- Project planning/storyboard sheet, one per group

- Poster paper or large white paper for the final course
- Optional: craft materials for theming (a fairy-tale retelling, an obstacle-course race, a theme park map, etc.)

Warm-Up (10 minutes)

Whole-class review: brainstorm everything learned across the unit onto chart paper — Color Codes, block programs, loops, conditionals. Introduce the project: teams will build one course that uses both marker Color Codes and a block program.

Project Brief

Each team designs a themed course where Ozobot: (1) begins with a short block program (for example, a starting "dance" using lights and sound, built with a sequence and at least one loop or conditional), then (2) follows a hand-drawn path containing at least three different Color Codes, and (3) finishes with a Win/Exit code or a final block-triggered celebration.

Planning (15 minutes)

1. Teams choose a theme and sketch their course layout on the planning sheet.
2. Teams list which three-plus Color Codes they will use and where, and outline the logic of their block program (sequence, loop, and/or conditional).

Build & Test (30–40 minutes)

1. Teams draw their final course on poster paper, following spacing rules for Color Codes.
2. Teams build their OzoBlockly program and calibrate it to Ozobot.
3. Teams run a full test from start to finish, debugging any codes or blocks that don't perform as planned, and iterate until the course works reliably.

Presentations (15–20 minutes)

Each team runs their course for the class and explains their theme, which Color Codes they chose and why, and what their block program does. Classmates may ask one question per presentation.

Assessment

- Uses at least three Color Codes correctly and purposefully
- Block program includes a clear sequence plus at least one loop or conditional
- Team can explain the reasoning behind their design choices
- Course runs successfully after debugging, or team can explain what they would fix with more time

Differentiation

- Support: offer a menu of pre-themed course templates with a simpler required code/block count.
- Extension: teams add a second Ozobot for a timed "race" mode using Timer codes, or add a second conditional for more complex behavior.

Appendix A: Quick-Reference Color Code Chart

These are the Color Codes used most often across Lessons 1 and 2. For the complete official chart, including timers, counters, and win/exit codes, download the free printable guide from files.ozobot.com.

Color Code	Marker Sequence	What Ozobot Does
Line Switch Left	Green – Green – Red	Turns left at the next intersection
Line Switch Straight	Green – Green – Blue	Goes straight through the next intersection
Line Switch Right	Red – Red – Green	Turns right at the next intersection
U-Turn	Blue – Blue – Red	Turns around and follows the line back
U-Turn (end of line)	Red – Blue	Turns around when the line simply stops
Slow	Red – Red – Black	Slows Ozobot down for 3 seconds
Fast	Blue – Blue – Black	Speeds Ozobot up for 3 seconds
Turbo	Blue – Blue – Green	Ozobot's fastest speed for 3 seconds
Pause	Red – Red – Blue	Ozobot stops moving for 3 seconds

Drawing tips:

- Draw lines about the width of the Ozobot marker tip (roughly 1/4 inch).
- Leave a short blank stretch of plain black line (a "safe zone") on either side of a code so Ozobot can read it cleanly.

- Keep colors solid and fully colored in, without gaps or overlapping strokes.

Indiana Academic Standards—Aligned with the Ozobot Unit

Computer Science (Grades 3–5 band, adopted 2023)

Programs & Algorithms (PA)

- **3-5.PA.1** – Collaborate with peers to implement problem solving steps to create a variety of programming solutions → Lessons 2, 5 (group course design)
- **3-5.PA.2** – Using age-appropriate vocabulary, explain steps taken and choices made to improve the design of a sequence → Lessons 1–5 (all "explain your codes/program" moments)
- **3-5.PA.3** – Design programs that incorporate sequences, events, loops, and conditionals → the core standard for Lessons 3 and 4 (OzoBlockly loops/if-then)
- **3-5.PA.4** – Test and debug (identify and fix errors) a program or algorithm to ensure it runs as intended → every lesson's build-and-test step
- **3-5.PA.5** – Construct and analyze solutions to problems using the basic steps of algorithmic problem solving → Lesson 5 project design

Mathematics (Grade 4, Indiana Academic Standards)

Geometry

- **4.G.4** – Identify, describe, and draw rays, angles (right, acute, obtuse), and perpendicular and parallel lines using appropriate tools → Lessons 1–2 (drawing intersections and turns) and Lesson 5 (course design)
- **4.G.5** – Classify triangles and quadrilaterals based on the presence or absence of parallel or perpendicular lines, or the presence or absence of angles → useful if course shapes (squares, loops) are discussed geometrically

Measurement

- **4.M.1** – Measure length to the nearest quarter-inch, eighth-inch, and millimeter → Lessons 1–2 and 5 (ruler use when drawing lines/safe zones)
- **4.M.5** – Understand that an angle is measured with reference to a circle... and can be used to measure other angles → Lesson 3 (90° turn blocks in the square program)
- **4.M.6** – Measure angles in whole-number degrees using appropriate tools. Sketch angles of specified measure → Lesson 3 (planning turn angles for the square)

Algebraic Thinking

- **4.AT.6** – Describe a relationship between two variables and use to find a second number when a first number is given. Generate a number pattern that follows a given rule → Lesson 4 (loop blocks as repeated patterns)