

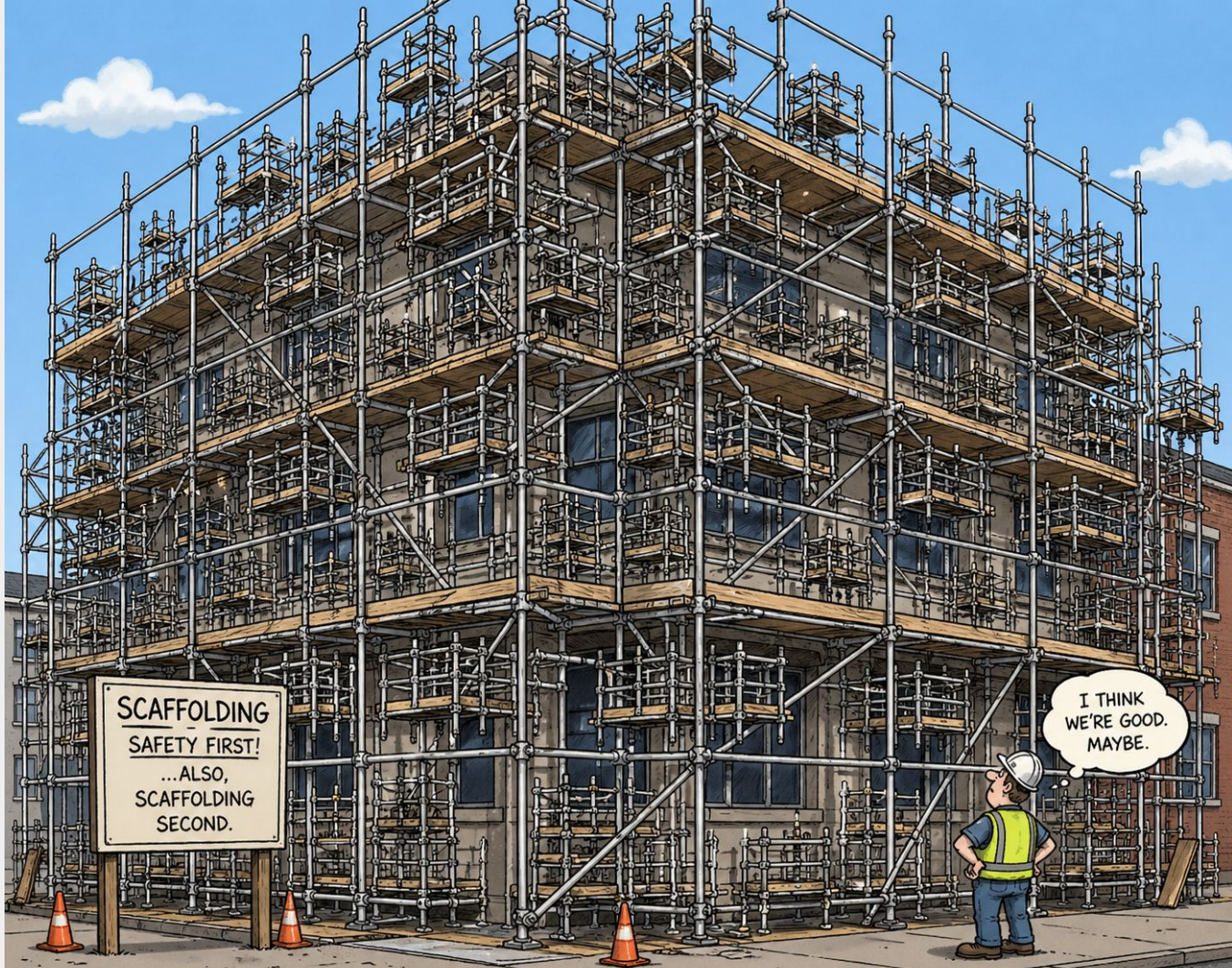
Seeing Isn't Believing: Collaborative Debugging with VEX AIM

Audra Selkowitz
Senior Education Developer



Workshop Goals

- Learn about how bugs in a project can be powerful learning tools
- Build your troubleshooting toolbox
- Practice debugging ALM projects together
- Create your own bugged projects



SCAFFOLDING

SAFETY FIRST!

...ALSO,
SCAFFOLDING
SECOND.

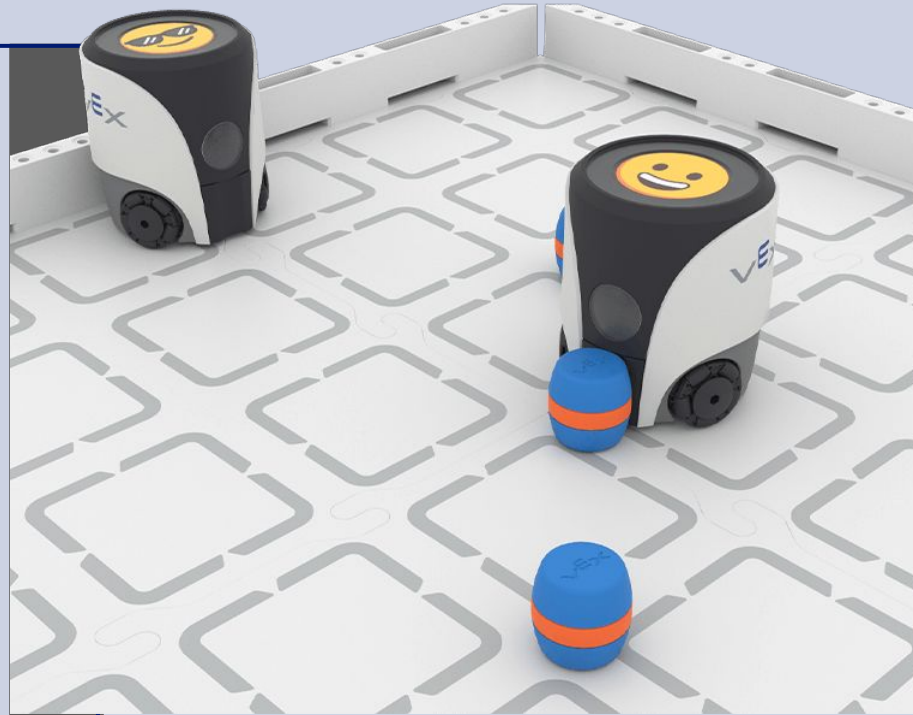
I THINK
WE'RE GOOD.
MAYBE.



I'VE GOT
THE LADDER.
YOU'RE GOOD.

A Paradigm Shift...

“ ...bugs not as accidents to be solved but as objects-to-think-with and objects-to-share-with.”



A Paradigm Shift...

**“ The idea of designing bugs for learning
—or *debugging by design*—
makes learners agents of their own
learning and, more importantly, of
making and solving mistakes.”**

[-Yasmin B. Kafai](#)

“Mischievousness and fun as well as empathy and sensitivity...were **productive emotions** exhibited during bug design, and **shifts away from frustration to increased comfort, security, and a sense of control with bugs** were expressed retrospectively weeks afterward.”

21st-Century Skills

Foundational Literacies

How students apply core skills to everyday tasks

1. Literacy
2. Numeracy
3. Scientific literacy
4. ICT literacy
5. Financial literacy
6. Cultural and civic literacy

Competencies

How students approach complex challenges

7. Critical thinking/ problem-solving
8. Creativity
9. Communication
10. Collaboration

Character Qualities

How students approach their changing environment

11. Curiosity
12. Initiative
13. Persistence/ grit
14. Adaptability
15. Leadership
16. Social and cultural awareness

Lifelong Learning

Bugs support creativity, resiliency, and learning.

What do you need to know to debug a project?

- What is the intention of the project?
- How are you trying to do that?
- Where does it go wrong?



Find the Bug!

```
when started
  get all cargo ▾ data from AI Vision
  if AI Vision object is sports ball ▾ ? then
    set [robot icon] LED color to yellow ▾
  else if AI Vision object is orange barrel ▾ ? then
    set [robot icon] LED color to red ▾
  else if AI Vision object is blue barrel ▾ ? then
    set [robot icon] LED color to blue ▾
  else
    set [robot icon] LED color to off ▾
```

This project should make the robot show a different color for each object that is detected repeatedly.

when started

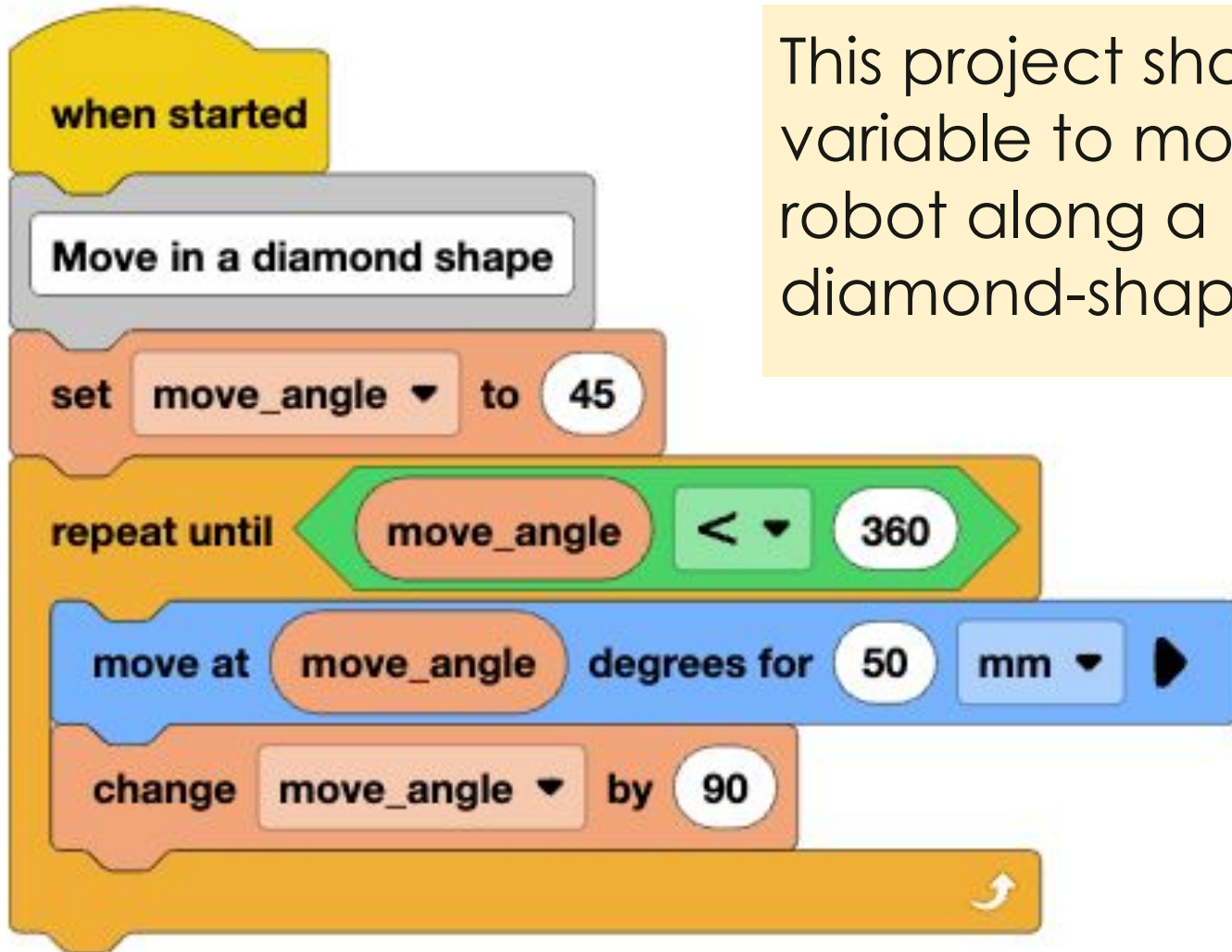
Move in a diamond shape

repeat 4

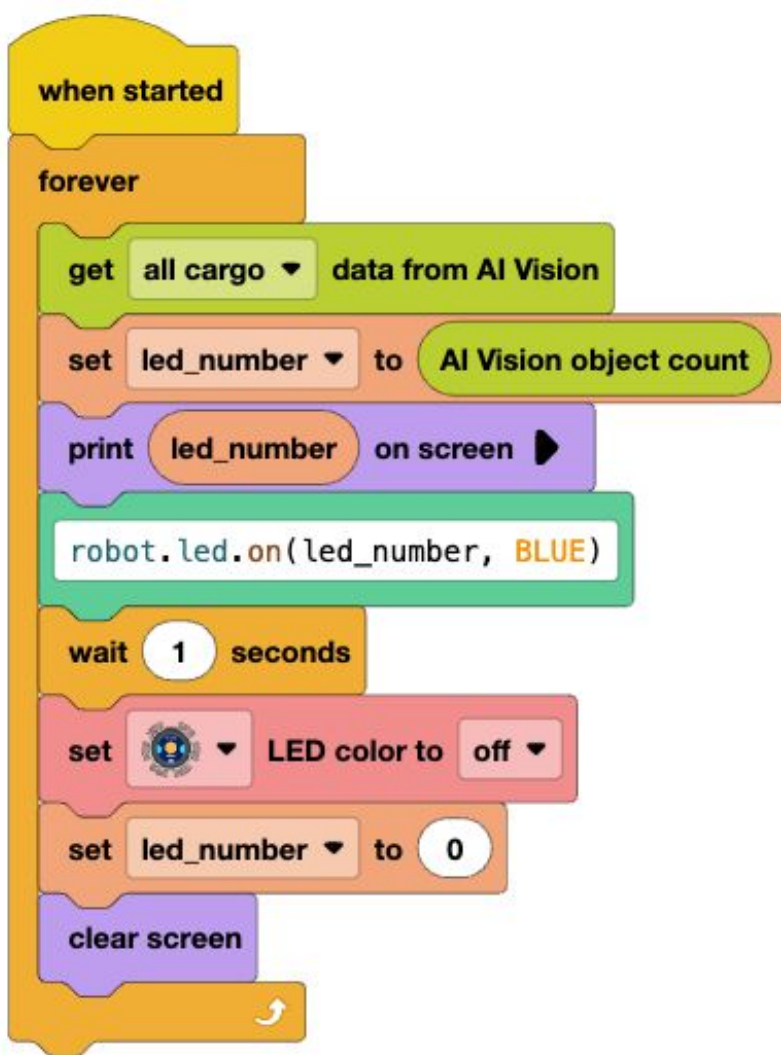
set my_variable ▼ **to** 45

move at my_variable **degrees for** 50 **mm** ▼ ▶

This project should use a variable to move the robot along a diamond-shaped path.



This project should use a variable to move the robot along a diamond-shaped path.



This project should detect how many objects are in view, and glow that number of LEDs.

The image shows a Scratch script designed to control an LED's color based on a user's input. The script starts with a 'when started' block, followed by an 'ask' block that prompts the user with 'What is your favorite color?' and waits for an answer. A series of 'if-then' blocks check if the answer contains the words 'red', 'green', or 'blue'. If a match is found, the 'set LED color to' block is configured with the corresponding color. An 'else' block at the bottom ensures that if the answer does not contain any of the specified colors, the LED is turned off.

Monitor

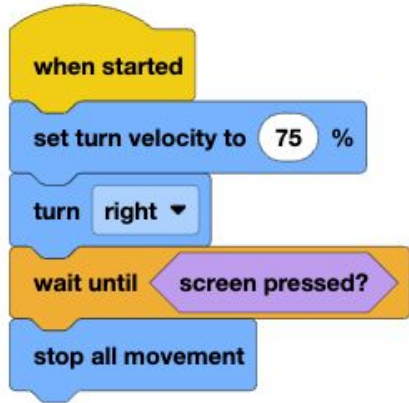


Vision Dashboard ▶

Console ▼

```
What is your favorite color?  
>>> purple
```

This project should glow the LEDs the same color as the answer given in the console.



```
when started
  set turn velocity to 75 %
  turn right
  wait until screen pressed?
  stop all movement
```

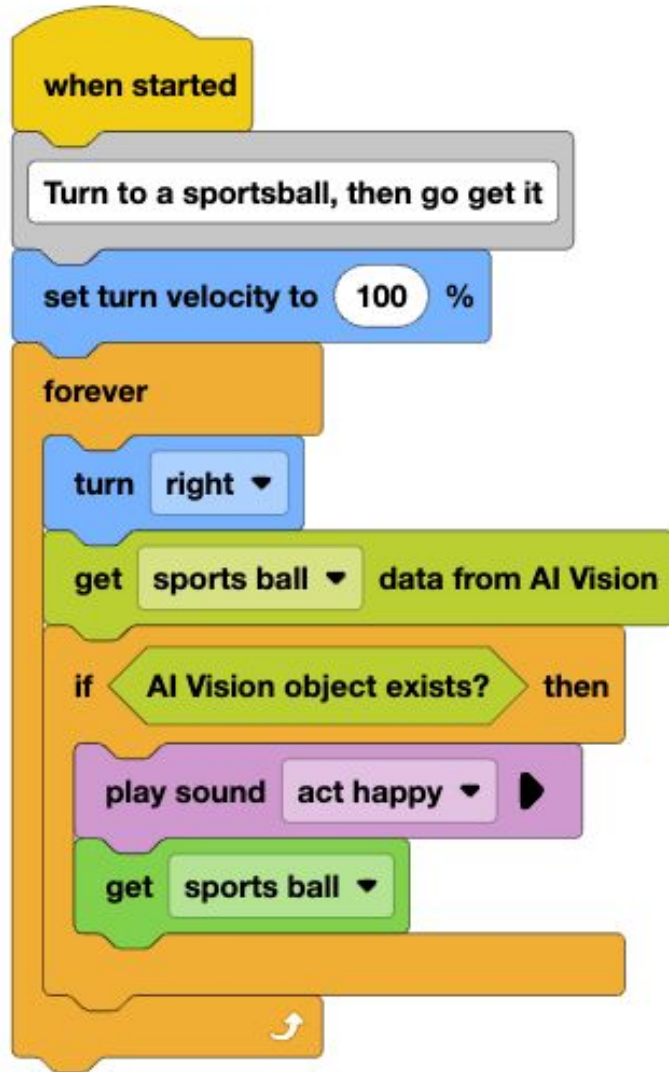
A Scratch script starting with a yellow 'when started' block. It is followed by a blue 'set turn velocity to 75 %' block, a light blue 'turn right' block, an orange 'wait until screen pressed?' block, and finally a blue 'stop all movement' block.



```
when screen pressed
  set LED color to blue
  wait 0.2 seconds
  set LED color to green
  wait 0.2 seconds
  set LED color to white
  wait 0.2 seconds
  set LED color to purple
  wait 0.2 seconds
  set LED color to cyan
  wait 0.2 seconds
  set LED color to off
```

A Scratch script starting with a yellow 'when screen pressed' block. It is followed by a sequence of eight blocks: a red 'set LED color to blue' block, a yellow 'wait 0.2 seconds' block, a red 'set LED color to green' block, a yellow 'wait 0.2 seconds' block, a red 'set LED color to white' block, a yellow 'wait 0.2 seconds' block, a red 'set LED color to purple' block, a yellow 'wait 0.2 seconds' block, a red 'set LED color to cyan' block, a yellow 'wait 0.2 seconds' block, and finally a red 'set LED color to off' block.

This project should make the robot turn continuously and use the LEDs to glow a disco pattern when the screen is pressed.



This project make the robot turn until it finds a sports ball, then act happy and go get it.



Motion

Motion

Motion - Actions



Emoji

move forward ▾



Kicker

move at 90 degrees



Sound

move forward ▾ for 200 mm ▾ ▶



LED

move at 45 degrees for 200 mm ▾ ▶

Message



Macro

turn right ▾

AI Vision

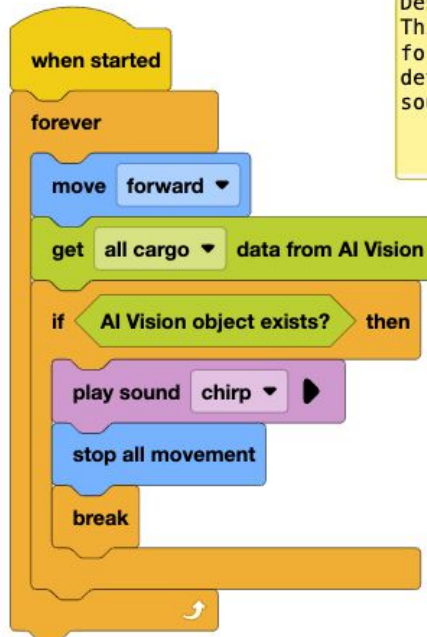


Screen

turn right ▾ for 90 degrees ▶

Project:
Object Detection Alert

Description:
This project will move the AIM robot forward until any cargo is found. When detected, the robot will play a beep sound, and stop all robot movement.



What did you notice?

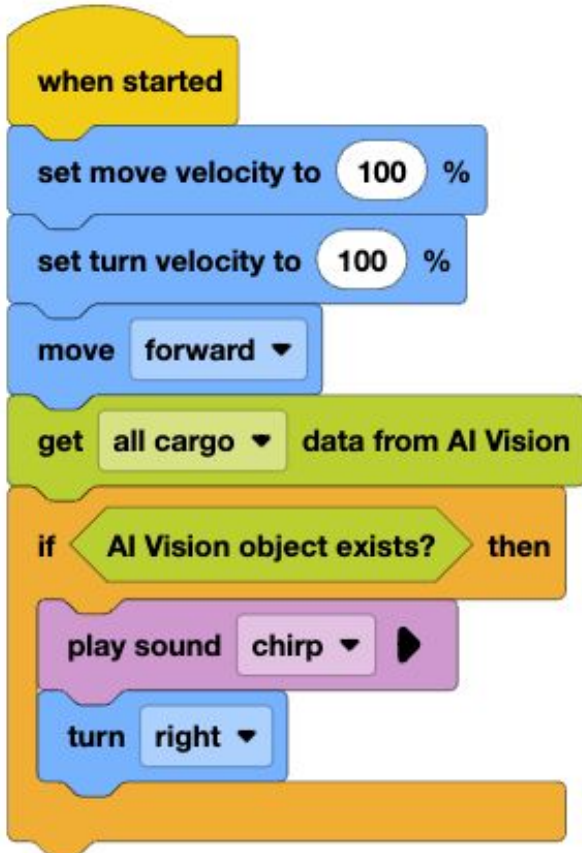
- How was this experience for you?
- How does it feel when you find a bug?
- Can we think about bugs in the same ways we think about challenges?



Bug “Buckets”

- Logic
- Sequence
- Technical
- How the robot works
- Listening/attention
- Pseudocoding
- Syntax
- Overcomplicating

Your turn!

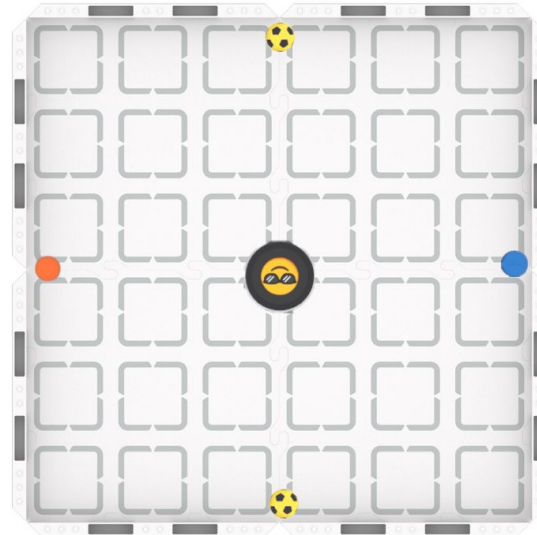


```
when started
  set move velocity to 100 %
  set turn velocity to 100 %
  move forward
  get all cargo data from AI Vision
  if AI Vision object exists? then
    play sound chirp
    turn right
```

The image shows a sequence of Scratch code blocks. It starts with a yellow 'when started' block, followed by two blue 'set velocity' blocks for 'move' and 'turn', both set to 100%. Then a blue 'move forward' block. A green 'get all cargo data from AI Vision' block. An orange 'if AI Vision object exists?' block with a 'then' tab. Inside the 'if' block are a purple 'play sound chirp' block and a blue 'turn right' block.

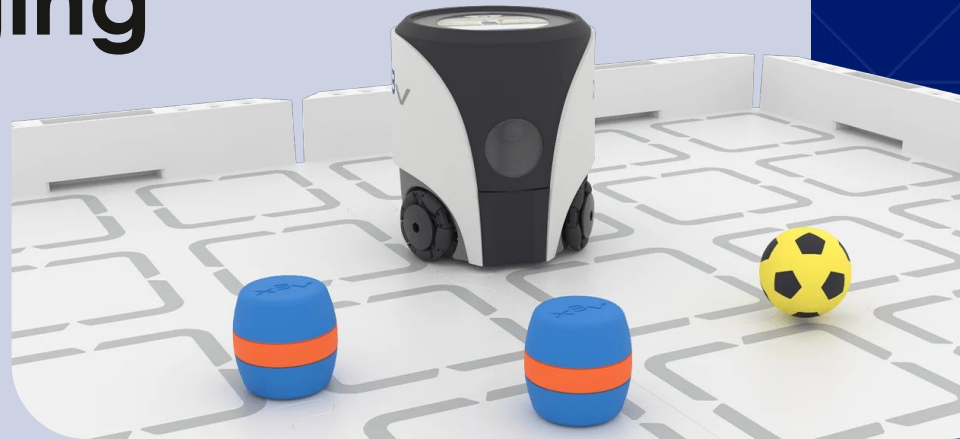
Project:
AI Vision navigation - cardinal directions

Description: The robot should move and turn to face each object, using AI Vision. It should play a sound when each object is detected.



Debugging Check-in

- What were the bugs?
- How did you fix them?
- How did debugging feel for you?



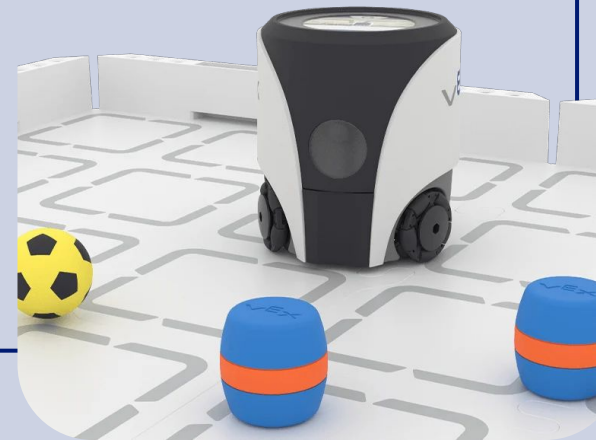
**Stump the
Software
Team!**

Create your own bugged project!

Create a project with at least 2-3 bugs in it.

Think creatively about the bugs you're making!

Use the bugged project or
create your own.



Plan Your Bugged Project

- **What is your robot supposed to do?**
- **How are you going to try to do that?**
- **Where/how is it going to go wrong?**
 - Think of the different “bug buckets” we identified. Which are you pulling from and why?
- **What are possible solutions for your bugs?**

Build Your Bugged Project

- **Where are the bugs?**
- **What are the solutions?**
 - Make sure you have an example of functioning code to be able to tell if our experts successfully “fixed” the bugs.

Bring This Experience to Your Students

- **Making bugs demonstrates understanding**
 - You have to know the concept in order to create a bug with it
- **Intention is important**
 - Their ideas are good – execution is the problem
 - Have to know what the coder was trying to do, to debug intentionally

Bring this experience to **your students**

- **There is more than 1 way to fix a bug**
 - Multiple solutions to the same problem helps us all learn
- **Creating bugs builds creativity and resilience**
 - The more you deal with bugs in non-threatening ways, the more persistent and comfortable you become in solving them

Stay Connected

Let's Connect!

Tag me in the **VEX PD+ Community!**

@Audra_Selkowitz

Want to Learn More? Join VEX PD+ as an All-Access Member!

Schedule a **1-on-1 Session** in VEX PD+