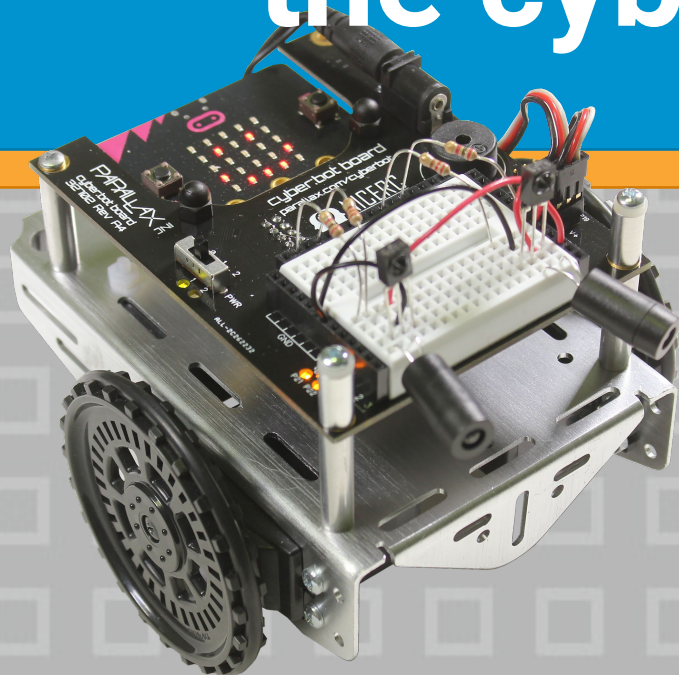


# Getting started with Python and Microbit using the cyber:bot Webinar



Hosted by **Ken Gracey**, Parallax CEO  
Co-host **Kevin Nolten**, NICERC Director of  
Academic Outreach





## Ken Gracey, Parallax CEO

Started Parallax's educational program around microcontrollers, electronics and robotics in 1997. Rocklin, California. Preliminary CTE credential in California. Enjoys unicycling, mountains and robots!

E-mail: [kgracey@parallax.com](mailto:kgracey@parallax.com)



## Kevin Nolten, NICERC Director of Academic Outreach

Runs Department of Homeland Security cyber education grant and creates cyber education curriculum with district/state level education departments. Has two funny dogs, loves outdoors and and is taking up powered paragliding.

E-mail: [kevin.nolten@cyber.org](mailto:kevin.nolten@cyber.org)



**NICERC**  
AN ACADEMIC DIVISION OF THE  
CYBER INNOVATION CENTER



# cyber:bot is a Joint Project of Parallax and NICERC



- <https://www.parallax.com>
- Established educational robotic company since 1992
- American manufacturer from Rocklin, California
- Staff of 20



- <https://nicerc.org/>
- Department of Homeland Security funded
- Create a cyber-ready workforce
- Produces a free curriculum for American educators
- Staff of 25



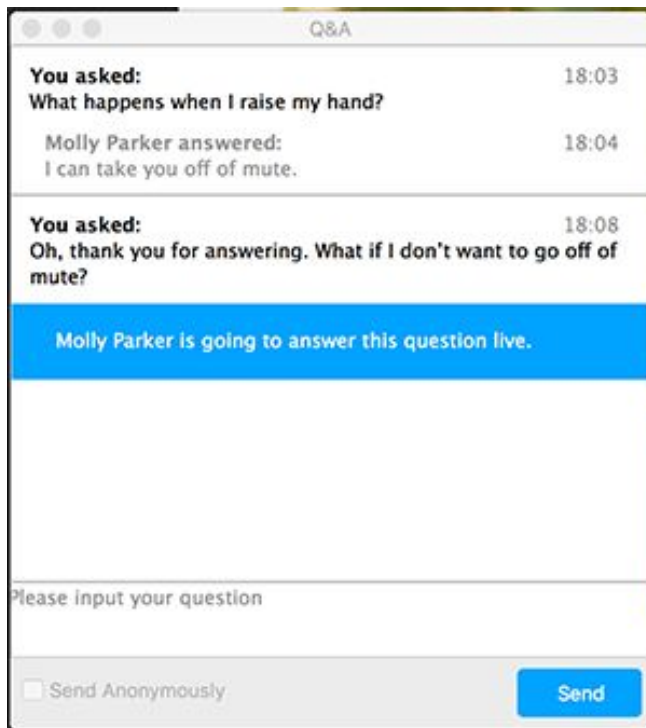
# Webinars are interactive - we want your questions and comments!



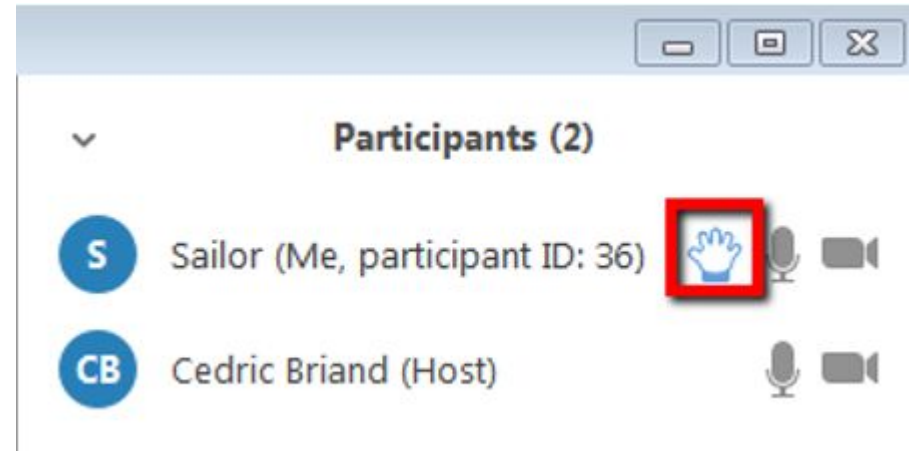


# How to use Zoom!

Use the Q&A to  
**TYPE** question,  
make a comment.



Raise your hand to **ASK** a question or make a comment. Your audio should be working to do this.



Lower your hand when you are done.



**NICERC**  
AN ACADEMIC DIVISION OF THE  
CYBER INNOVATION CENTER



# Agenda

- Microbit (5 min.)
  - Python language
  - Microbit from BBC in UK
  - Programming the Microbit
  - Educational resources
- cyber:bot (15 min.)
  - Adding Cyberbot module
  - Program the cyber:bot for turns
  - Curriculum
  - Educator's Resources
  - Cybersecurity - **new!**
- Demonstrations! (20 min.)
- Support and Products (5 min.)
- Q/A (15 min.)

# Guest Goals

- Enjoy the webinar event
- Learn about the resources
- See some Python
- Identify setup challenges
- Understand the cyber:bot module and hardware interaction
- Gain confidence to use cyber:bots in classrooms
- See robots GO
- Get questions answered



**NICERC**  
AN ACADEMIC DIVISION OF THE  
CYBER INNOVATION CENTER



# Parallax Manufactures in the USA





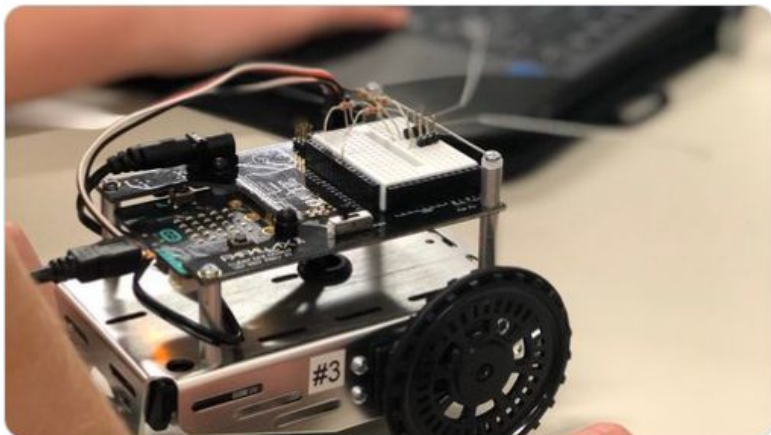
# Cyber:bots Success One Year Anniversary

You Retweeted



**Mitch Miller** @MMiller\_32 · Nov 8, 2019

Today my 7th grade class completed their #CyberBot challenges. We programmed these bots with Python script and guided them through floor obstacles. Thank you @ParallaxInc and @CIC\_NICERC for supplying the robots for our students!



1

4

17

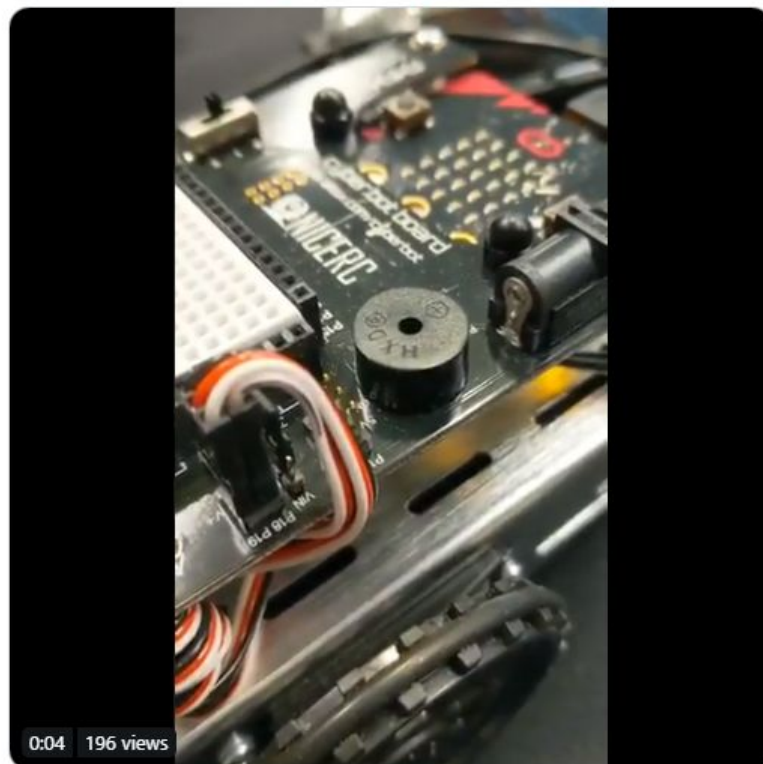


You Retweeted



**Shaun Langevin** @ShaunLangevin · Nov 14, 2019

I think I coded this cyber:bot with a sequence of appropriate tones. Great set of sessions at #vtfest2019. @mmuusdvt @ParallaxInc @CIC\_NICERC #vted



0:04 196 views

1

5

9



**NICERC**  
AN ACADEMIC DIVISION OF THE  
CYBER INNOVATION CENTER



# Ask a student what they think.

ZakUak (subscribe to his YouTube) spent two days at the Parallax booth at the USA Science and Engineering Festival. He says about cyber:bot in his video:

- “No black box - the real thing working with components!”
- “This makes coding interesting - Python is a fun programming language”

Exposure creates careers and drives the economy.



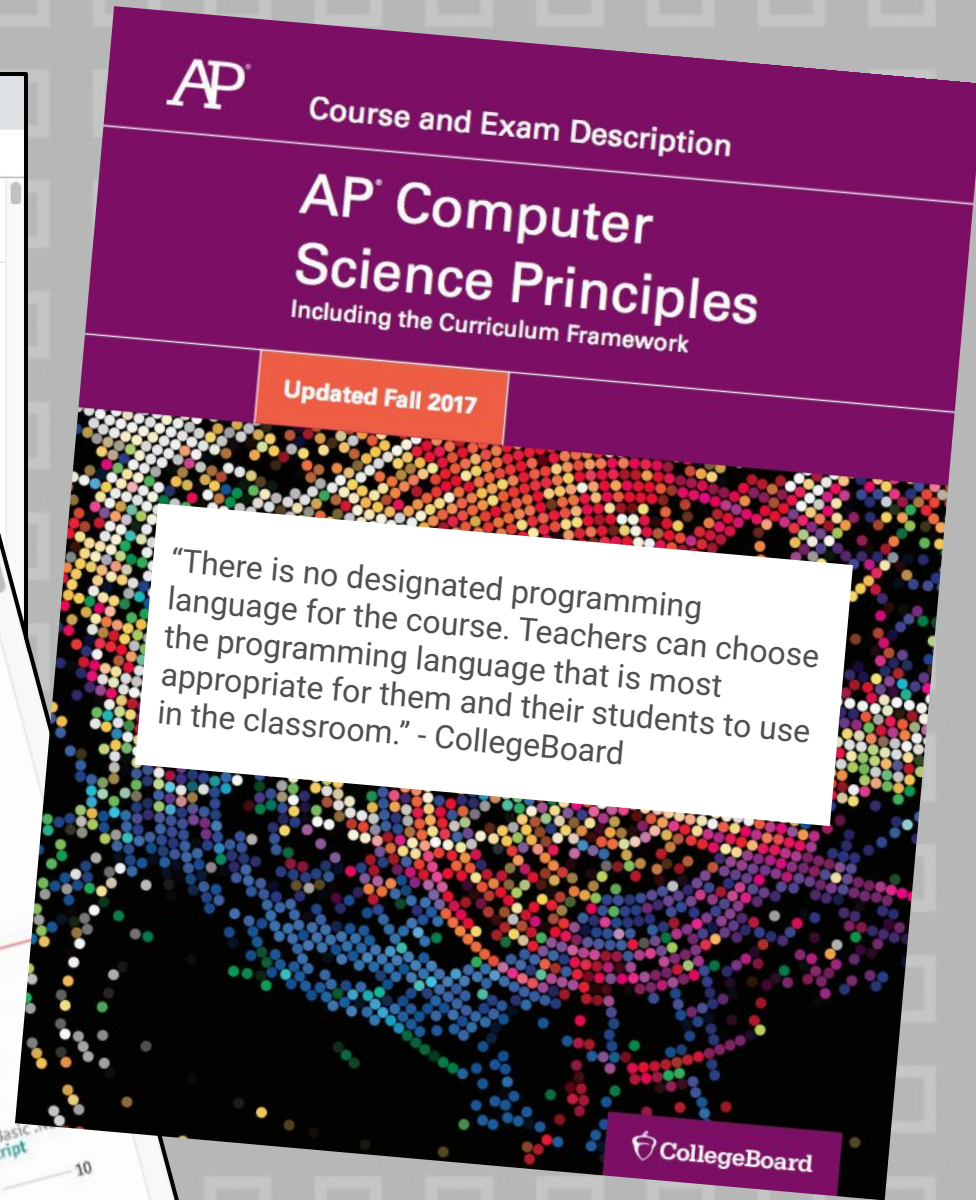
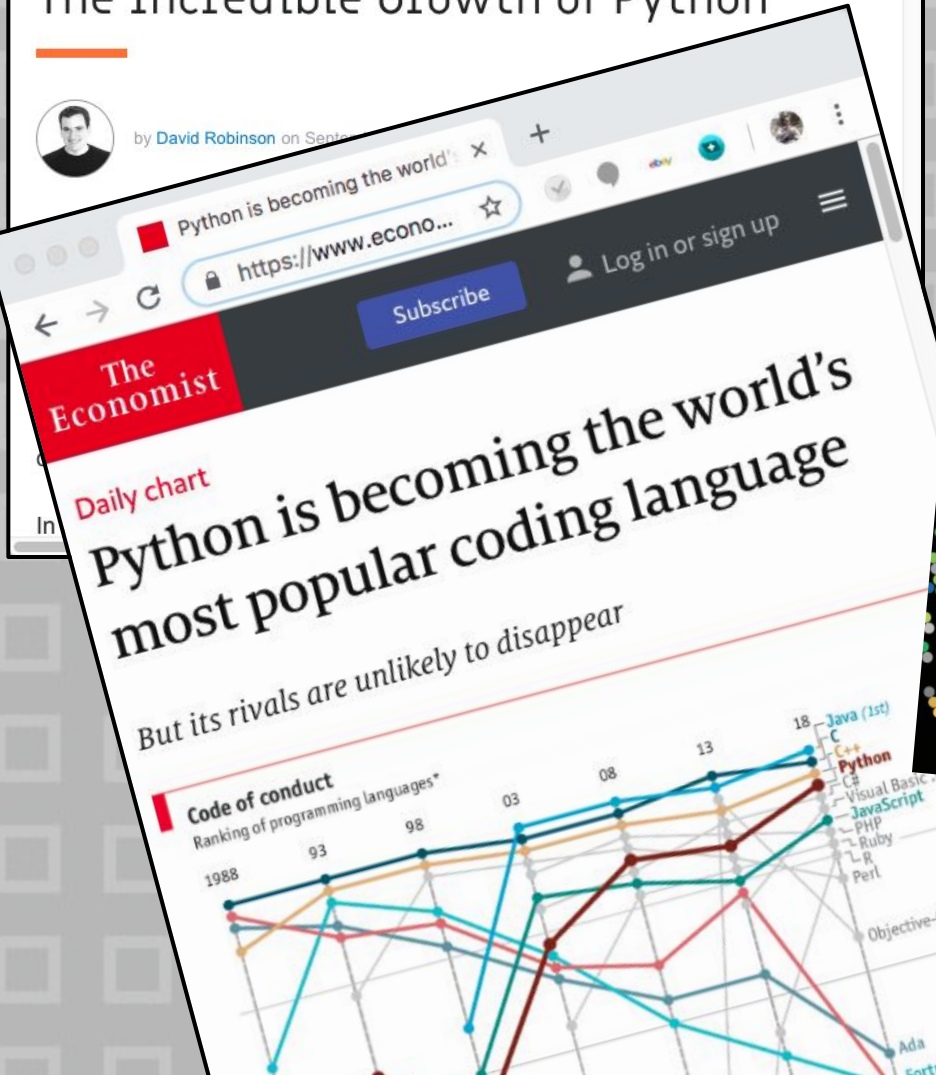
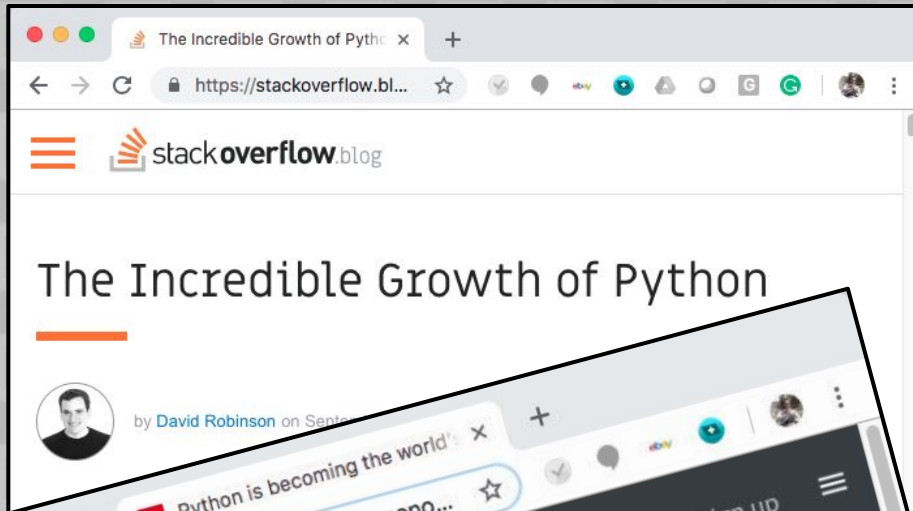
**NICERC**<sup>TM</sup>  
AN ACADEMIC DIVISION OF THE  
CYBER INNOVATION CENTER





# Python and micro:bit





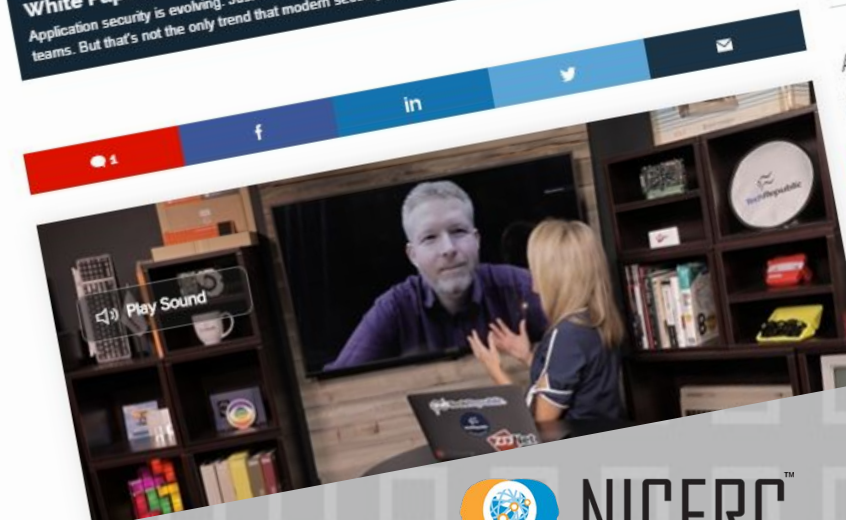
# Python is eating the world: How one developer's side project became the hottest programming language on the planet

Frustrated by programming language shortcomings, Guido van Rossum created Python. With the language now used by millions, Nick Heath talks to van Rossum about Python's past and explores what's next.

By Nick Heath | July 11, 2019 -- 14:58 GMT (07:58 PDT) | Topic: Developer

**Recommended Content:**  
**White Papers: Web Application Protection for the Modern Era - A Guide**  
Application security is evolving. Just the shift to primarily web-facing applications alone is enough to wag the system of many security teams. But that's not the only trend that modern security teams have to worry about. Introduce continuous...

[Download Now](#)



**RECOMMENDED FOR YOU**

A Guide to AppSec in the Age of APIs & Microservices  
White Papers provided by ThreatX

[DOWNLOAD NOW](#)

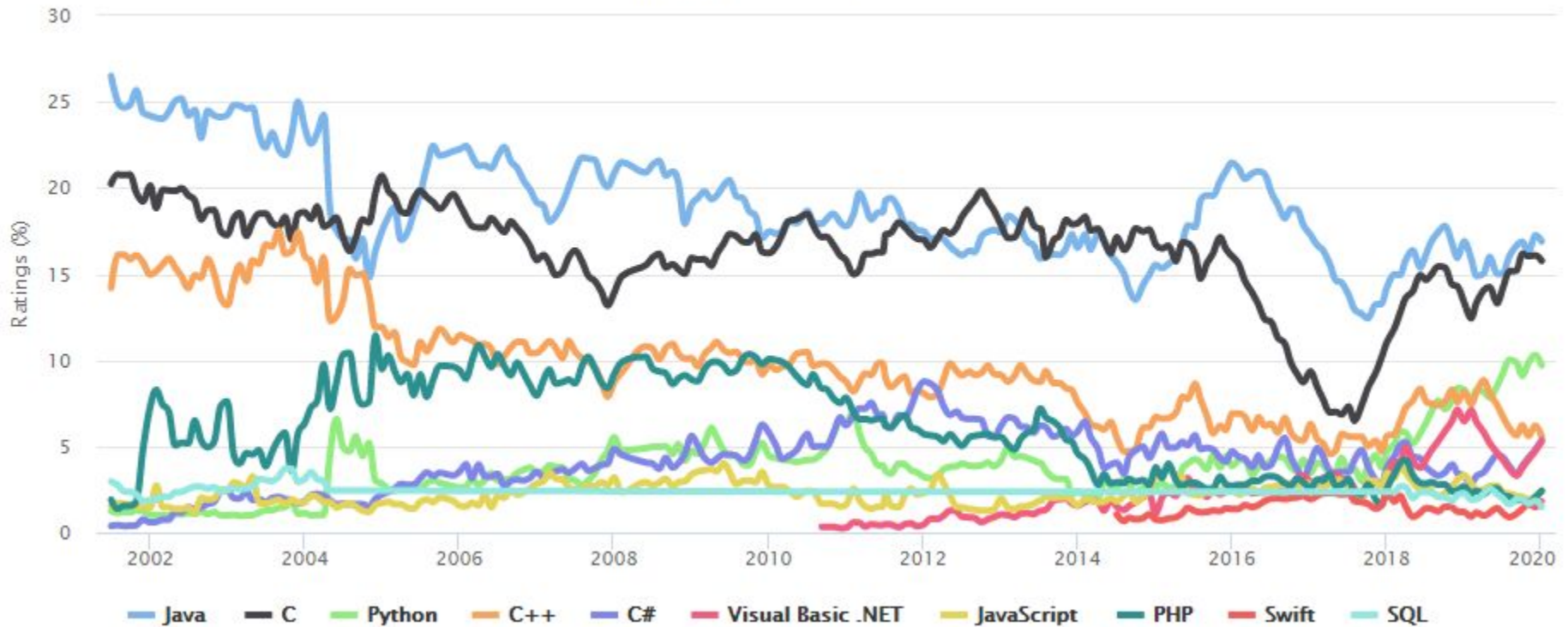
**MORE FROM NICK HEATH**

Windows 10  
Microsoft blocks major Windows 10 update for Surface Book 2 after bug makes GPU vanish



# TIOBE Programming Community Index

Source: [www.tiobe.com](http://www.tiobe.com)



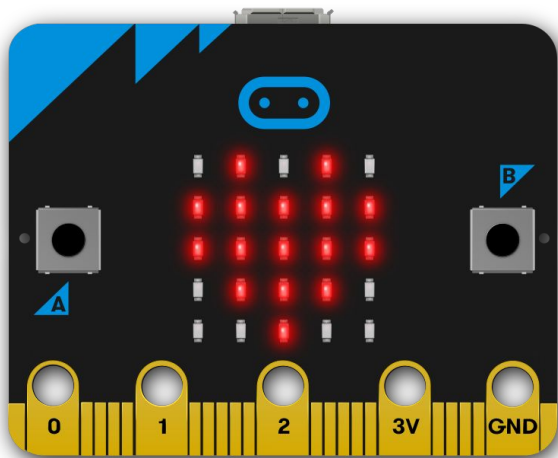
Jan 2020	Jan 2019	Change	Programming Language	Ratings	Change
1	1		Java	16.896%	-0.01%
2	2		C	15.773%	+2.44%
3	3		Python	9.704%	+1.41%





- Beginner-friendly (reads like English - looks BASIC)
- Object-oriented, structured language
- Runs on embedded, physical hardware (MicroPython)
- Tons of examples freely available
- Programming tool support (open source, all OSs, etc.)
- Forces new programmers to use alignment/indentation for legibility (good practice)
- Not overly verbose - easier to "get at the heart" of the concept you're teaching (no wading through a bunch of meaningless syntax rules that obscure the instructional intent)

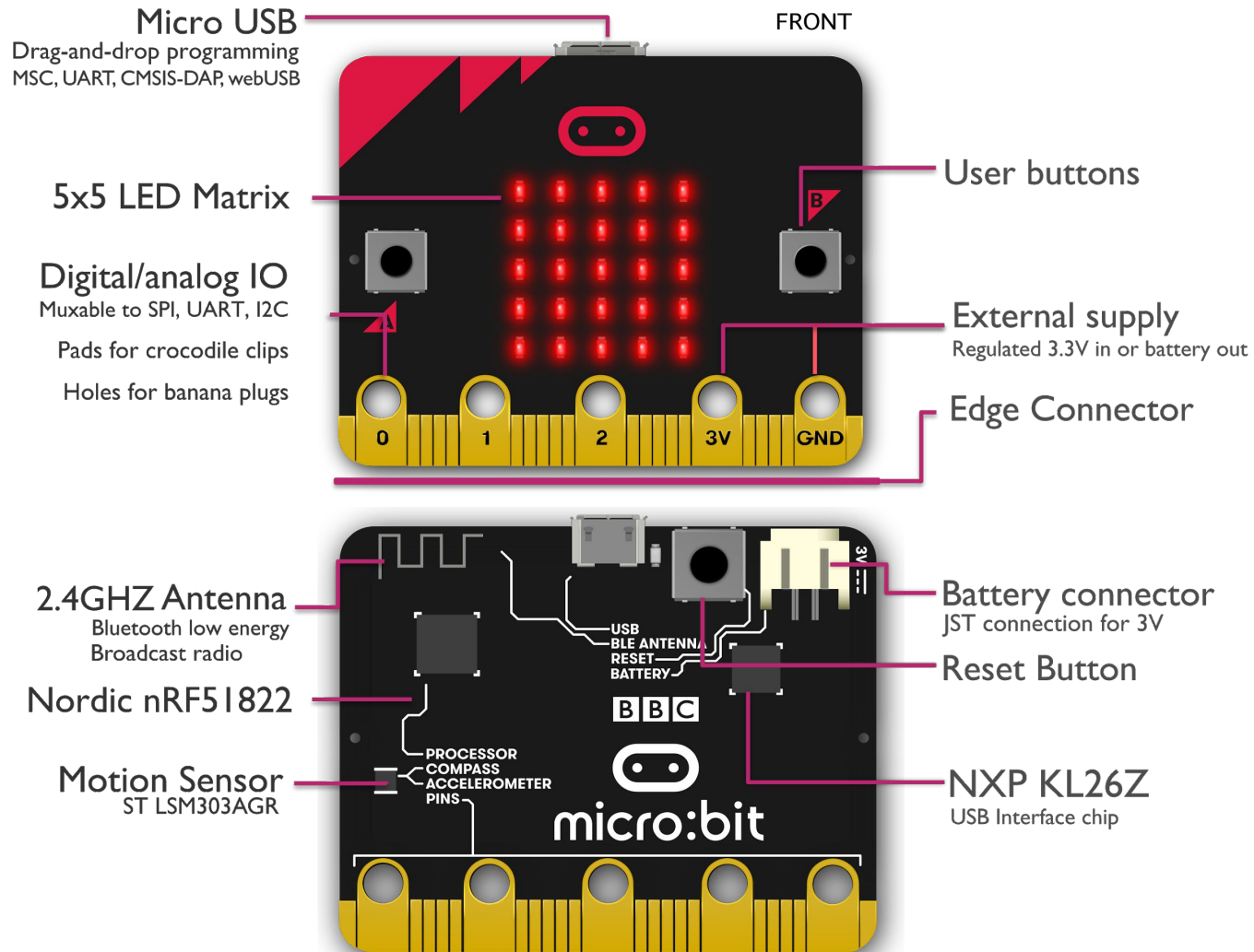
# micro:bit



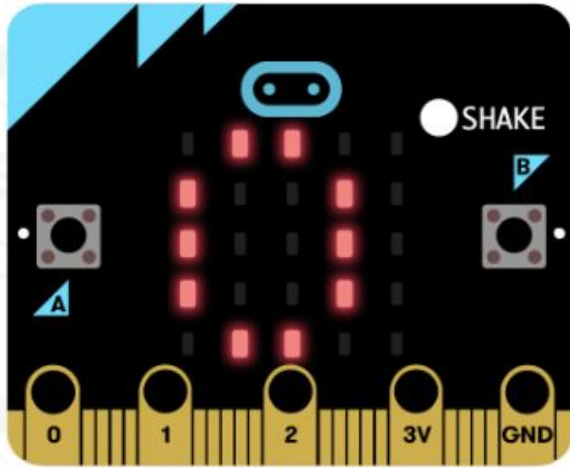
**NICERC**<sup>™</sup>  
AN ACADEMIC DIVISION OF THE  
CYBER INNOVATION CENTER



# micro:bit: Hardware Features







Search...



Basic

Input

Music

Led

Radio

Loops

Logic

Variables

Math

Advanced

on start

show string "Hello!"

show number 0

on shake ▾

clear screen

if Random ▾ = ▾ 2 then

show string "YES"

else if Random ▾ = ▾ 1 then ▾

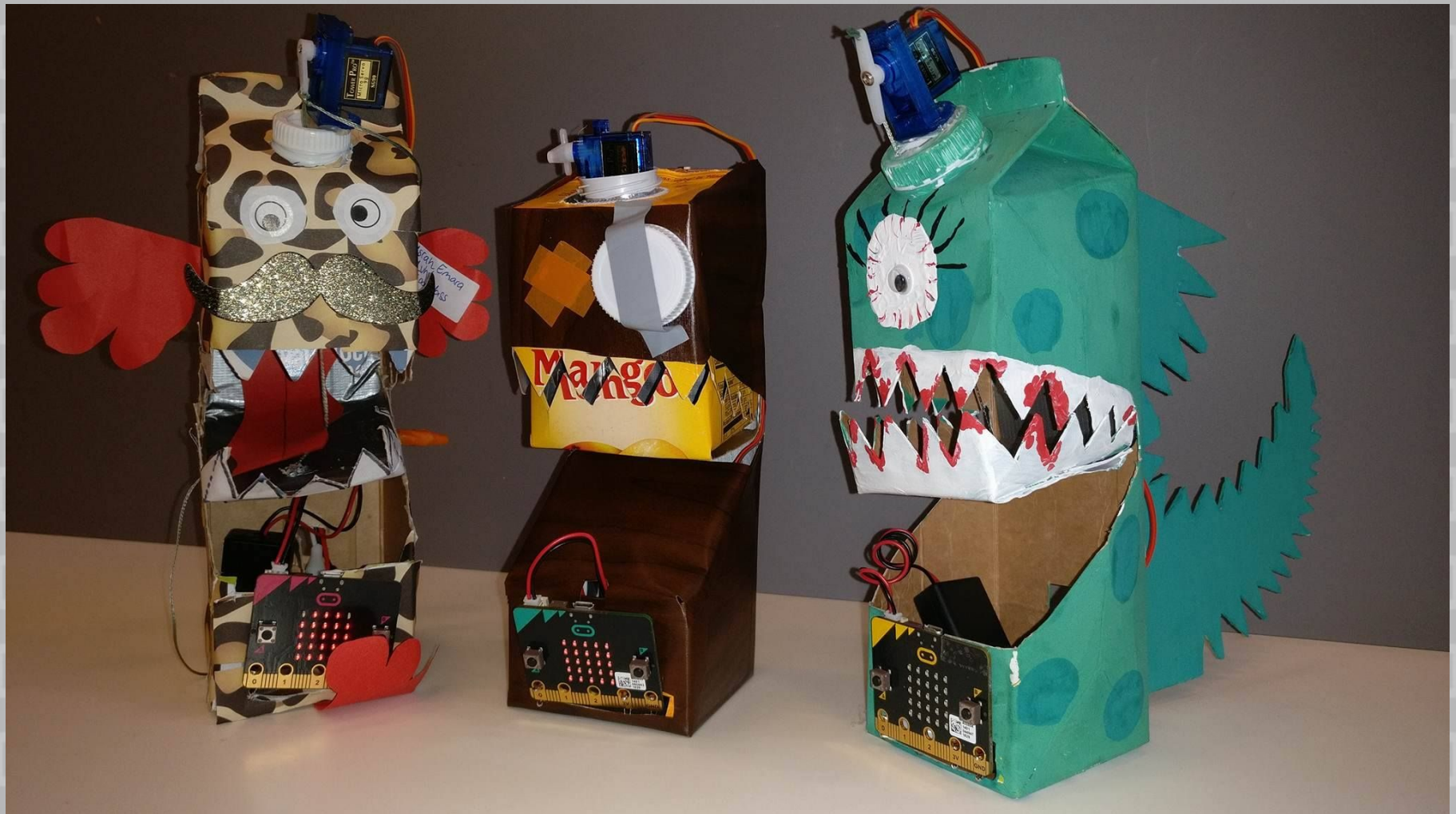
show string "NO"

else ▾

show string "I DON'T KNOW"

+

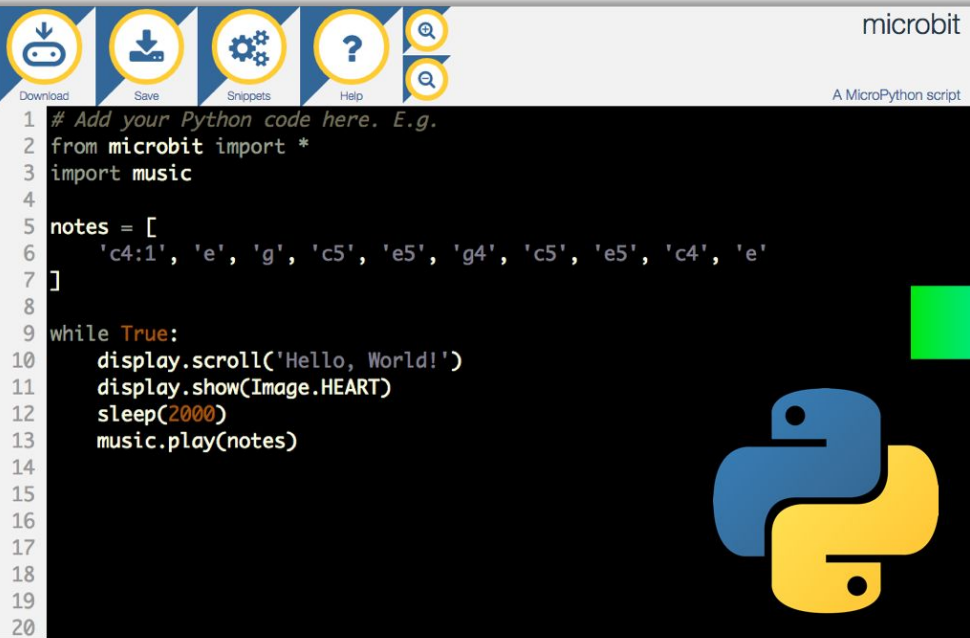




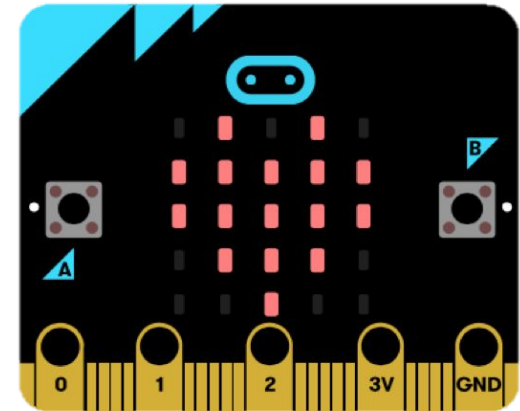
**NICERC**<sup>™</sup>  
AN ACADEMIC DIVISION OF THE  
CYBER INNOVATION CENTER



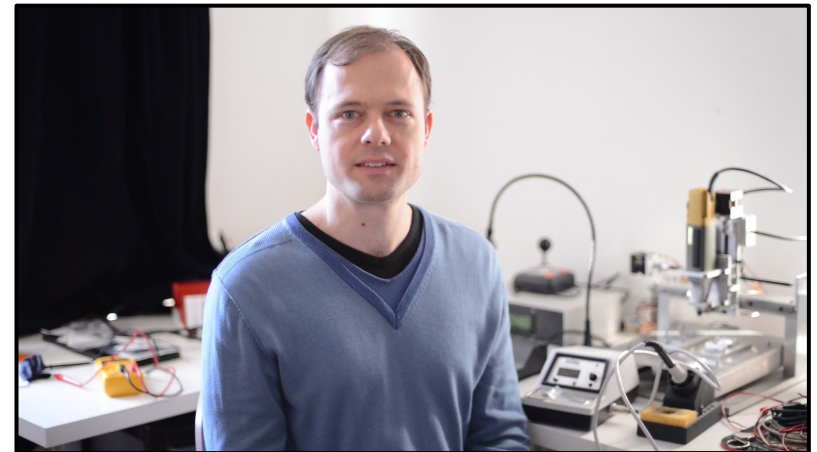
# MicroPython: Python for Microcontrollers



```
1 # Add your Python code here. E.g.
2 from microbit import *
3 import music
4
5 notes = [
6     'c4:1', 'e', 'g', 'c5', 'e5', 'g4', 'c5', 'e5', 'c4', 'e'
7 ]
8
9 while True:
10     display.scroll('Hello, World!')
11     display.show(Image.HEART)
12     sleep(2000)
13     music.play(notes)
14
15
16
17
18
19
20
```

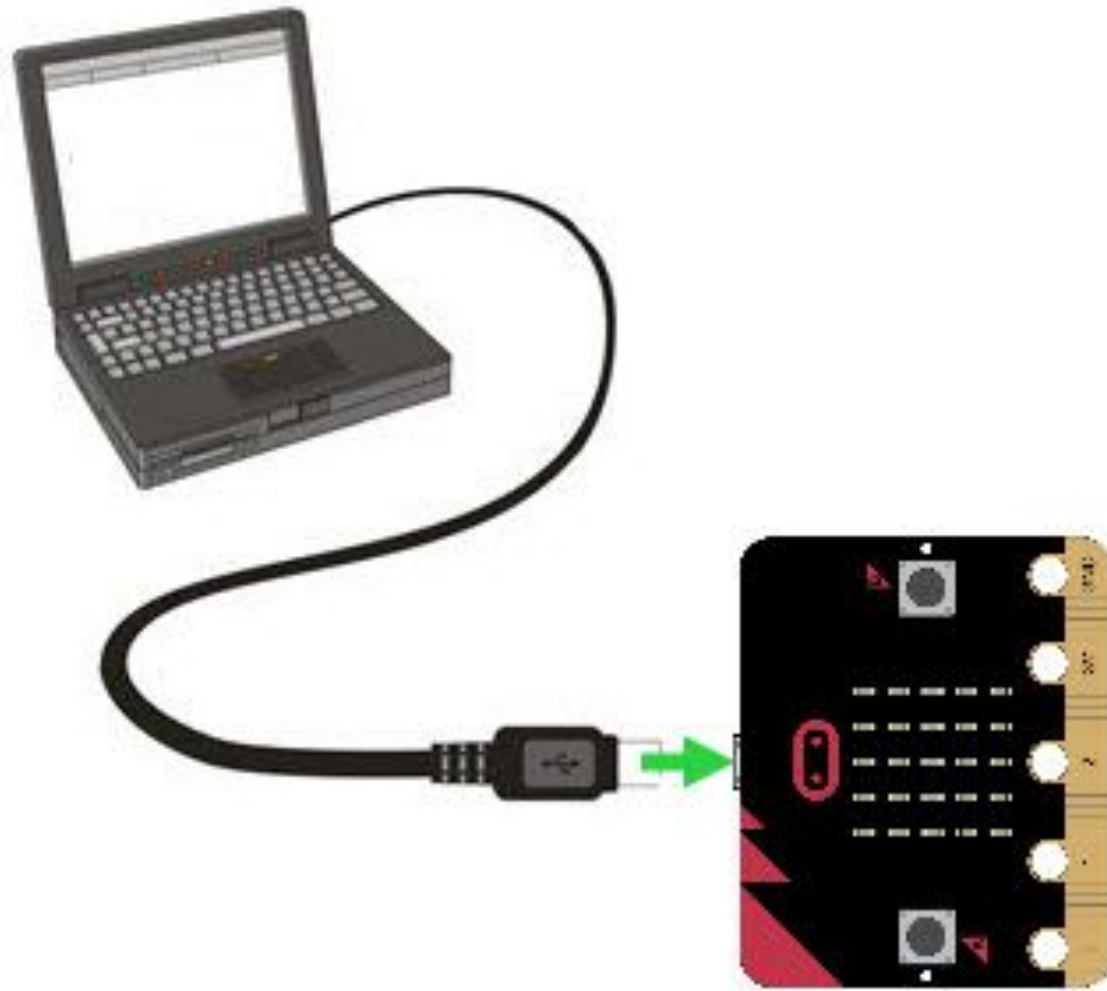


MicroPython is a creation  
of Damien George

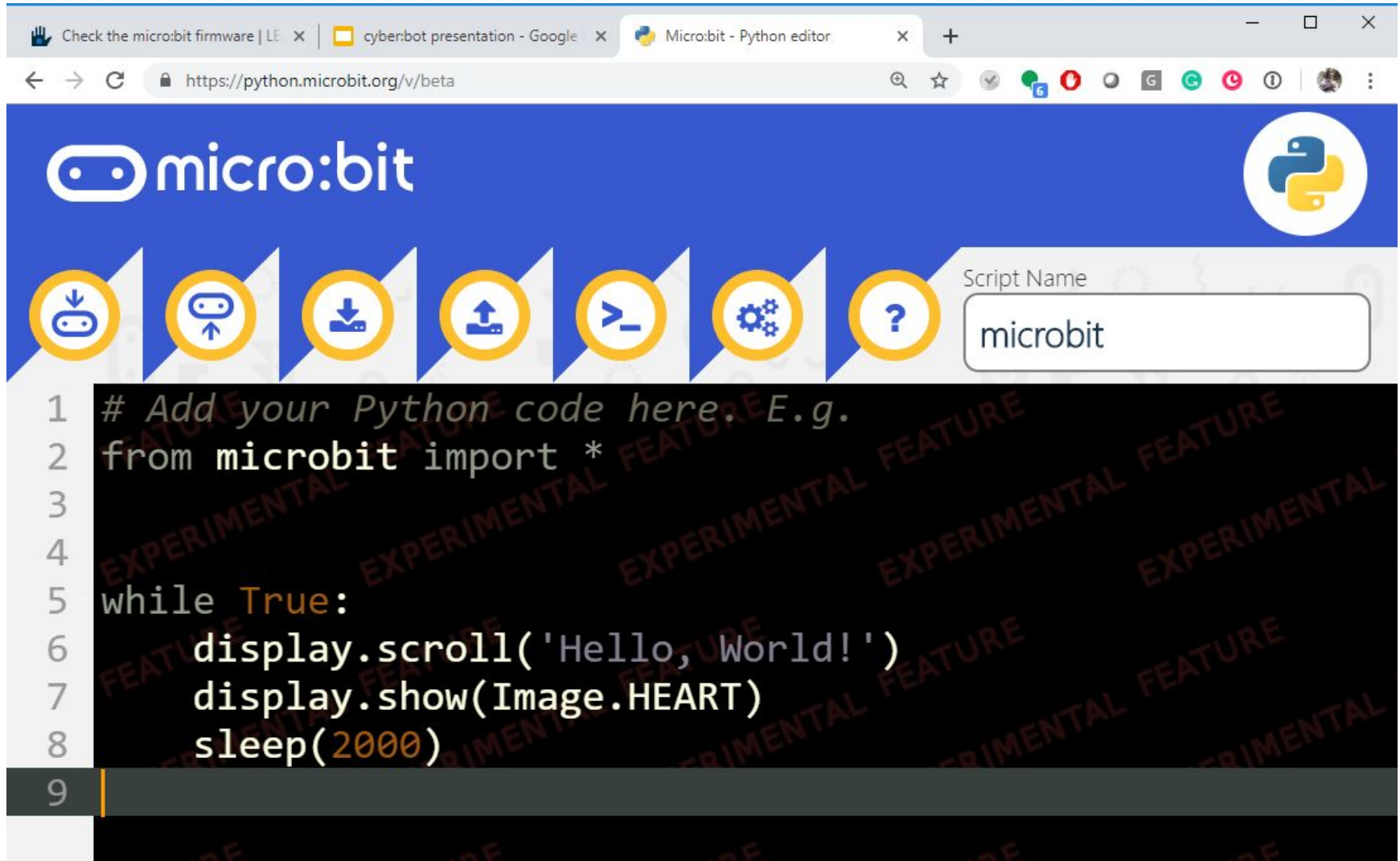




# Connect micro:bit to Computer



# Go to [python.microbit.org](https://python.microbit.org)



The screenshot shows a web browser with three tabs: "Check the micro:bit firmware | LE", "cyber:bot presentation - Google", and "Micro:bit - Python editor". The address bar shows the URL <https://python.microbit.org/v/beta>. The page header features the "micro:bit" logo on the left and a Python logo on the right. Below the header is a row of seven circular icons: a micro:bit, a micro:bit with an arrow, a download icon, an upload icon, a code icon, a settings icon, and a question mark. To the right of these icons is a "Script Name" input field containing the text "microbit". The main area of the page is a dark gray code editor with a light gray border. It contains the following Python code:

```
1 # Add your Python code here. E.g.  
2 from microbit import *  
3  
4  
5 while True:  
6     display.scroll('Hello, World!')  
7     display.show(Image.HEART)  
8     sleep(2000)  
9
```

# LED Matrix - Premade Images

```
#beating_heart.py  
  
from microbit import *  
  
display.show(Image.HEART)  
sleep (500)  
display.show(Image.HEART_SMALL)  
sleep (500)
```

- Press the “reset” button to see it again.
- Try your own - Google “micro:bit MicroPython Images”



# LED Matrix - Scrolling

```
#hello_goodbye.py
```

```
from microbit import *
```

```
display.scroll('Hello', delay = 500)
```

```
display.scroll('Goodbye', delay = 150)
```

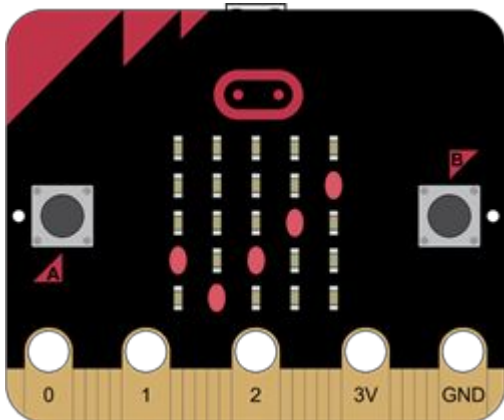
- Change the text with your own message.
- Change the delays.
- Add more lines of code and Flash!

# Pushbuttons

is\_pressed()

```
#is_pressed.py
from microbit import *

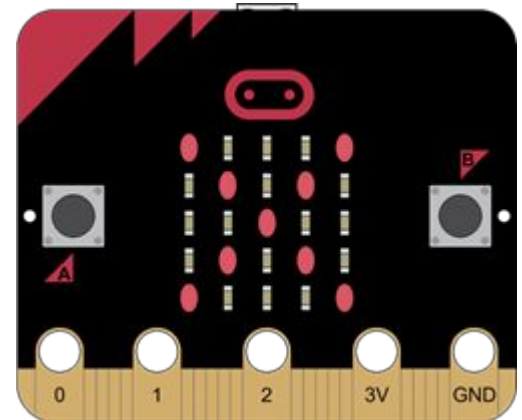
while True:
    if button_a.is_pressed():
        display.show(Image.YES)
    else:
        display.show(Image.NO)
```



was\_pressed()

```
#was_pressed.py
from microbit import *

while True:
    sleep(5000)
    if button_a.was_pressed():
        display.show(Image.YES)
    else:
        display.show(Image.NO)
```



**NICERC**  
AN ACADEMIC DIVISION OF THE  
CYBER INNOVATION CENTER



# Modules, Methods, Functions, and Objects

- *Modules* are code libraries that include the *objects*
- *Methods* are *functions* that belong to a specific *object*
- *Functions* are defined by a `def` statement. Functions can pass parameters (arguments) - like robot speed, sensor states.

```
#smile.py

from microbit import *

def smile():
    display.show(Image.HAPPY)
    sleep(2000)

smile()
```

microbit **module** ←

smile **function** ←

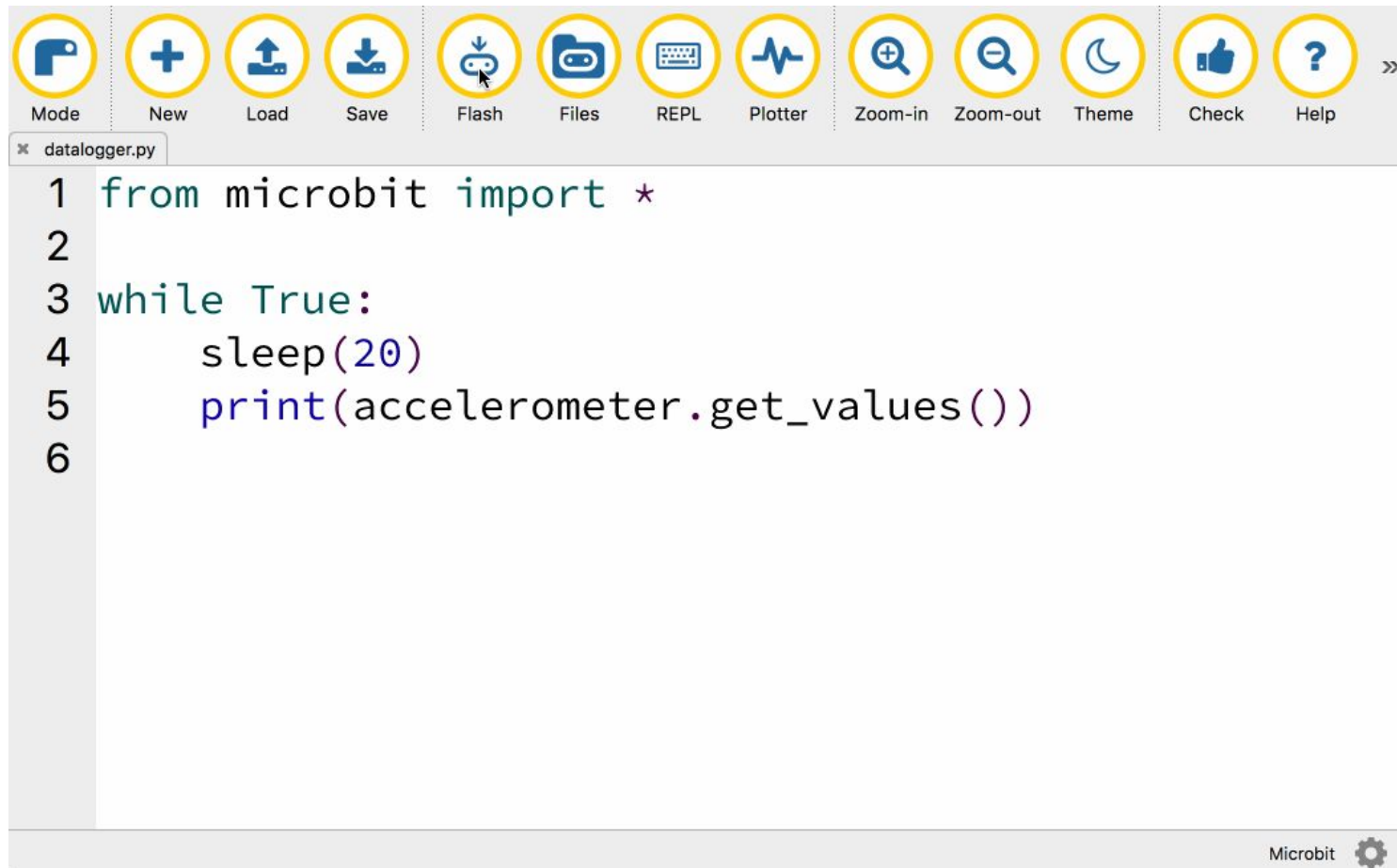
display **object** ←

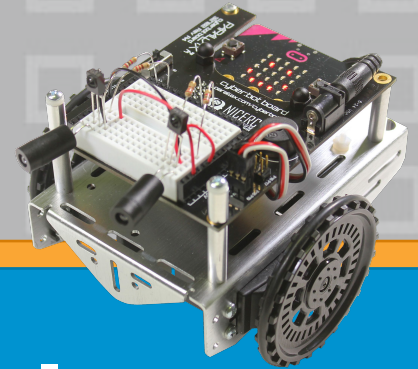
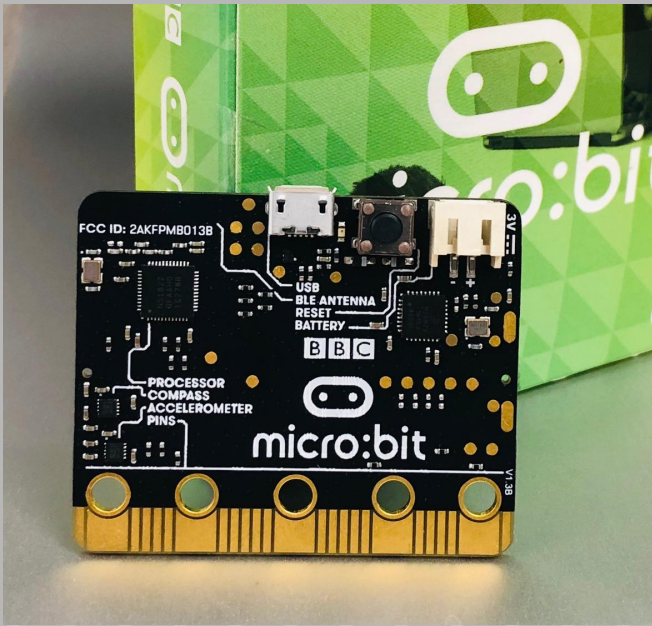
→ show **method**  
belongs to  
display **object**



# Alternative editor: Code with Mu

<https://codewith.mu/>





# Micro:bit Educational Resources

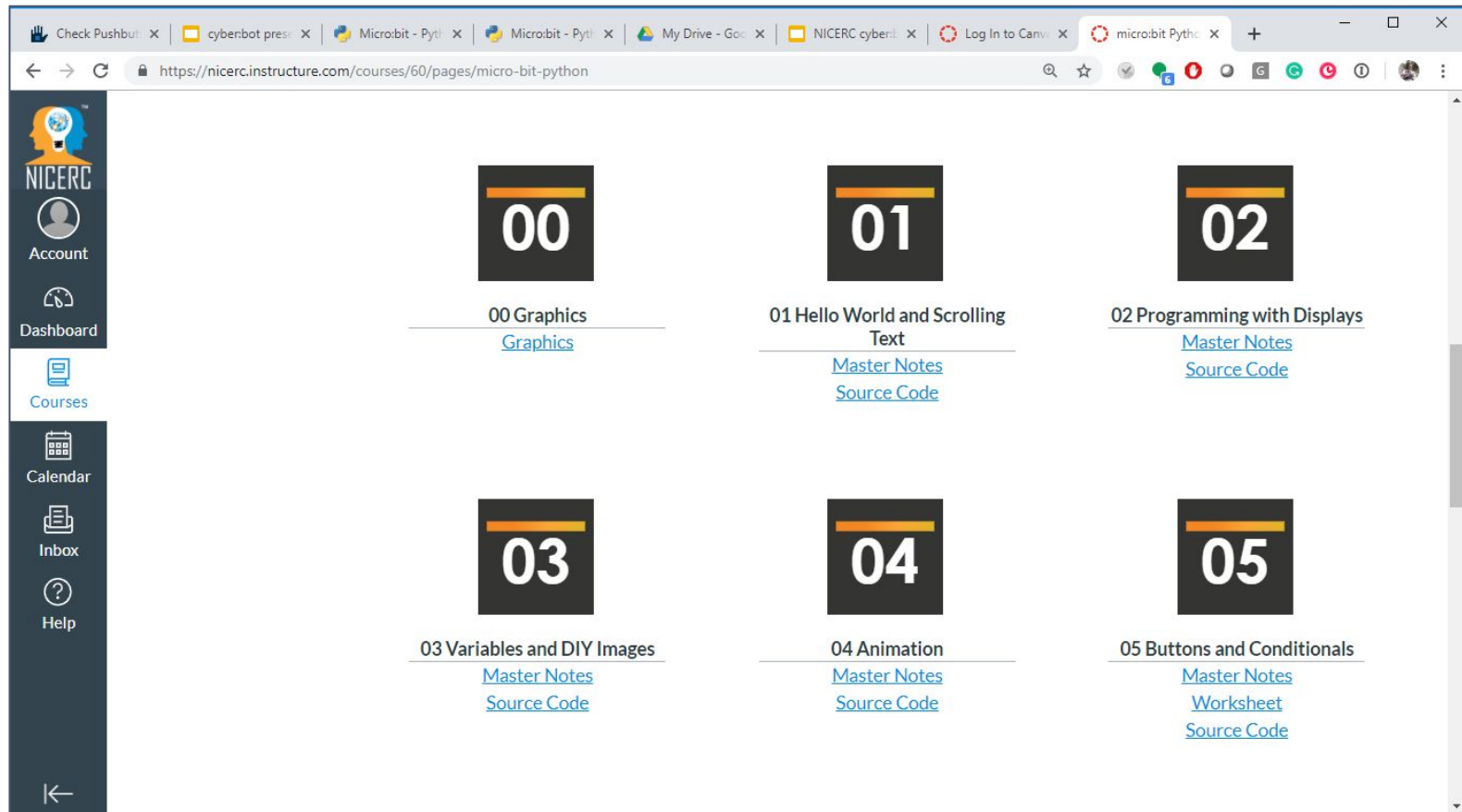


**NICERC**  
AN ACADEMIC DIVISION OF THE  
CYBER INNOVATION CENTER



# NICERC.org Cyber Fundamentals

Educators may request access at [nicerc.org](https://nicerc.org)



The screenshot shows a web browser window with the URL <https://nicerc.instructure.com/courses/60/pages/micro-bit-python>. The page features a dark sidebar on the left with icons for Account, Dashboard, Courses, Calendar, Inbox, and Help. The main content area displays six course modules arranged in a 2x3 grid. Each module is represented by a black square with a white number (00-05) and a title. Below each title are links for Master Notes and Source Code. The modules are:

- 00 Graphics: [Graphics](#)
- 01 Hello World and Scrolling Text: [Master Notes](#), [Source Code](#)
- 02 Programming with Displays: [Master Notes](#), [Source Code](#)
- 03 Variables and DIY Images: [Master Notes](#), [Source Code](#)
- 04 Animation: [Master Notes](#), [Source Code](#)
- 05 Buttons and Conditionals: [Master Notes](#), [Worksheet](#), [Source Code](#)



# NICERC.org Cyber Fundamentals



Lesson	Title
00	Graphics
01	Hello World and Scrolling Text
02	Programming with Displays
03	Variables and DIY Images
04	Animation
05	Buttons and Conditionals
06	Compass and Comparisons
07	Binary and Visual Counter
08	Tug-o-War
09	Communications
10	Student Response System
11	Passwords and Security
12	Voltage Measurement
13	Temperature Sensor

micro:bit 02 - Programming With Displays www.NICERC.org

**Teacher Notes:**

**Materials List**

**Per Group**

- Microbit and micro-USB cord
- Computer with access to the internet

**Objectives**

- Identify components on the Microbit
- Create code for Microbit using a Python editor
- Define the functions `display.set()` and `display.show()`
- Use x-y coordinates to reference grid of LEDs on Microbit

**LESSON NOTES**

- Start the lesson by reviewing the parts of the Microbit with students or giving a short quiz on the vocabulary and parts of the Microbit from the previous lesson.
- Remind students of the important line of code that should be included at the top of every program they create for the Microbit.  

```
from microbit import *
```
- Review `display.show()`. Demonstrate the display of the heart in lesson 01 and look at some of the other images that are built in to the micropython programming language. Specifically look at `Image, CHESSBOARD, Image, HOUSE, and Image, XMAS` as students will be recreating these images later in the lesson.
- This lesson focuses on a command called `display.set(x, y, i)`. This command is used to change the light levels of individual pixels. When using `display.set(x, y, i)`, students must specify three arguments: the x-coordinate, the y-coordinate, and the intensity level. The x and y coordinates range from 0 to 4, starting with 0,0 in the top left corner of the Microbit display. The intensity levels range from 0 to 9.

This material is based upon work supported by the U.S. Department of Homeland Security under Grant Award Number: 2013-PJ-127-000001, Modification #2.

Copyright © 2018 Cyber Innovation Center All Rights Reserved. Not for Distribution.

cyberbot11 Tuning Maneuvers Master Notes www.NICERC.org

**Teacher Notes:**

**Materials List**

- cyberbot11 PowerPoint - Tuning Maneuvers
- Assembled cyber bots

**Objectives**

- Illustrate the importance of finely tuned maneuvering functions.
- Finalize the understanding that the cyber bot servo only operate between the servo\_speak() range of 75 and 75.

**Notes**

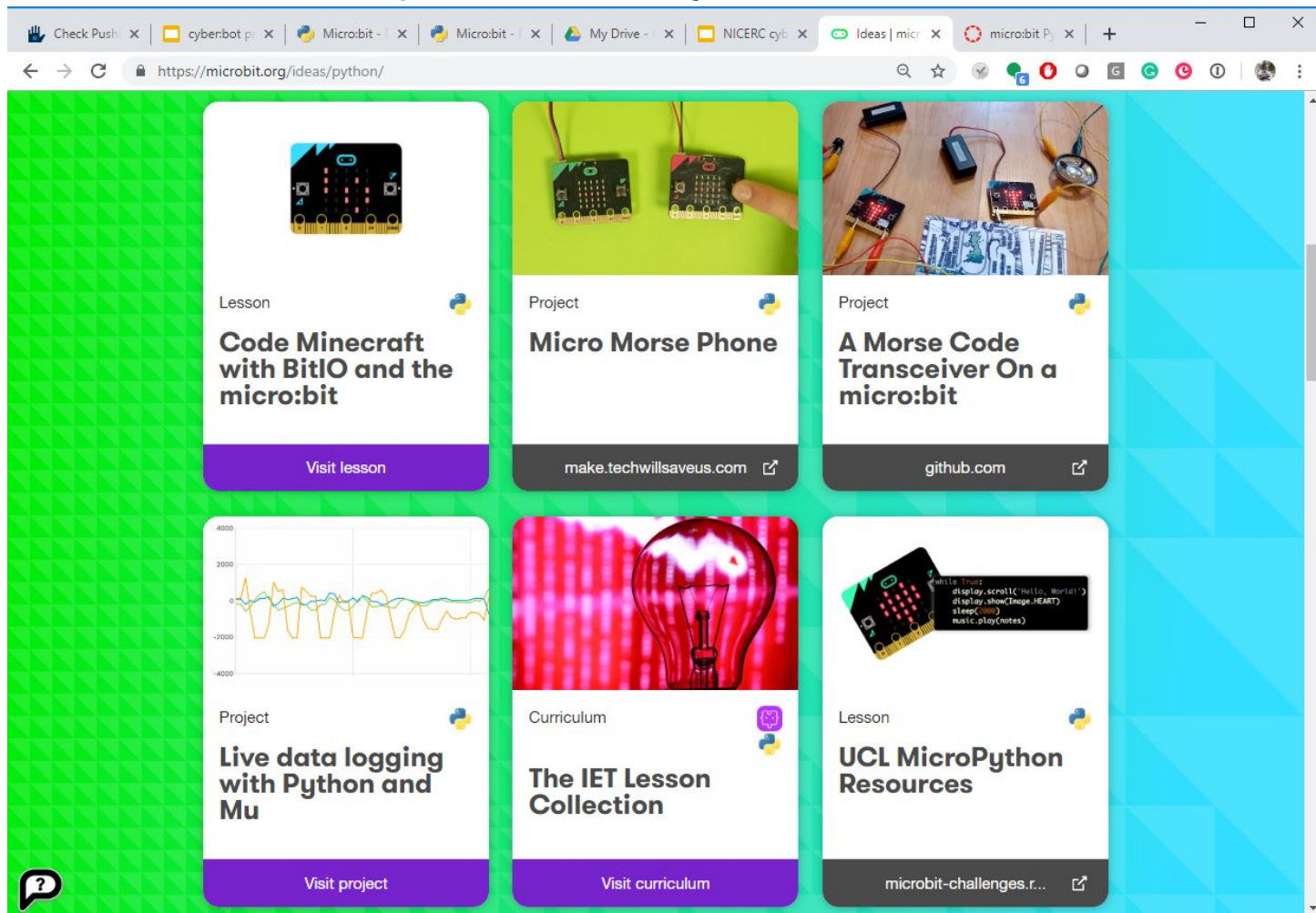
- The lesson opens with a review of some programming basics.
- The first question does not have an answer posted. Regarding the syntax, for example the function for "forward" must be labeled as "def forward()" to call the "forward" function, the command spelling and case must match exactly followed by (). The function allows the programmer to call upon a segment of code as often as they want by simply referring to the function label.
- The first activity will take the four functions created in the last lesson and "name" them. In other words, make the straighter straighter and the turns sharper.
- The slides take the class through the exploration of identifying how to correct "waving" forward and backward functions.
- Students will learn how to "comment" out a line of code. By commenting out a line of code, the code will not be uploaded to the cyber bot and therefore will not run. When testing the maneuvering functions, it will be important to test only one function at a time. To do this without starting a new program or without deleting the other functions, the programmer needs to only place a pound sign (#) at the beginning of a line that they want to comment out.

This material is based upon work supported by the U.S. Department of Homeland Security under Grant Award Number: 2013-PJ-127-000001, Modification #2.

Copyright © 2019 Cyber Innovation Center All Rights Reserved. Not for Distribution.

# micro:bit “Ideas”

<https://microbit.org/ideas/python/>



# BBC micro:bit Python Documentation

<https://microbit-micropython.readthedocs.io/en/latest/>

The screenshot shows a web browser with multiple tabs open, including 'Check Pushb...', 'cyberbot pre...', 'Microbit - Py...', 'My Drive - Go...', 'NICERC cyber...', 'BBC micro:bit', and 'microbit Pyth...'. The address bar shows the URL <https://microbit-micropython.readthedocs.io/en/latest/>. The page content includes a sidebar with a search bar and a list of tutorials: Introduction, Hello, World!, Images, Buttons, Input/Output, Music, Random, Movement, Gestures, Direction, Storage, Speech, Network, Radio, and Next Steps. The main content area is titled 'BBC micro:bit MicroPython documentation' and includes a 'Welcome!' message, an introduction to the BBC micro:bit and MicroPython, and a link to the 'First Steps with MicroPython' tutorial by Mike Rowbitt. The tutorial preview shows a photo of Mike Rowbitt, a diagram of the BBC micro:bit, and a code snippet: 

```
from microbit import *
# Edit your code here!
display.scroll("Hello, World!")
```

Check Pushb... x cyberbot pre... x Microbit - Py... x Microbit - Py... x My Drive - Go... x NICERC cyber... x BBC micro:bit x microbit Pyth... x +

← → ↻ 🔒 <https://microbit-micropython.readthedocs.io/en/latest/> ☆ 📧 🔔 ⚙️

**BBC micro:bit MicroPython**  
latest

Search docs

TUTORIALS

- Introduction
- Hello, World!
- Images
- Buttons
- Input/Output
- Music
- Random
- Movement
- Gestures
- Direction
- Storage
- Speech
- Network
- Radio
- Next Steps

API REFERENCE

- micro:bit MicroPython API

Read the Docs v: latest ▾

Docs » BBC micro:bit MicroPython documentation [Edit on GitHub](#)

## BBC micro:bit MicroPython documentation

Welcome!

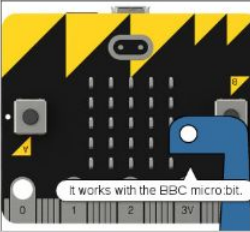
The BBC micro:bit is a small computing device for children. One of the languages it understands is the popular Python programming language. The version of Python that runs on the BBC micro:bit is called MicroPython.

This documentation includes lessons for teachers and API documentation for developers (check out the index on the left). We hope you enjoy developing for the BBC micro:bit using MicroPython.

If you're a new programmer, teacher or unsure where to start, begin with the tutorials.

### First Steps with MicroPython by Mike Rowbitt

MicroPython was created by Damien...



it works with the BBC micro:bit.

```
from microbit import *
# Edit your code here!
display.scroll("Hello, World!")
```

Everything you need to know about MicroPython on the BBC micro:bit is found in this documentation.

**Sponsored** Beat Triplebyte's coding quiz. Get offers from top companies. Skip resumes & recruiters. ads served ethically



# Hackster.io Educator Resources

## The Hardware (1), JavaScript (2) and Python (3)

### Micro:bit Basics for Teachers Part 1 - The Hardware

Are you a teacher who wants to use micro:bit in your classroom, but doesn't know where to start? We'll show you how!

 Beginner  Protip  30 minutes  3,291

#### Overview

Things

Story

Introduction

What is a micro:bit

How do I set up my micro:bit

Microbit Parts and Features

Lights

Buttons

Compass

Accelerometer

Pins

Bluetooth

Temperature Sensor

Credits

Comments (1)

 22







hackster.io

Micro:bit Basics for Teachers



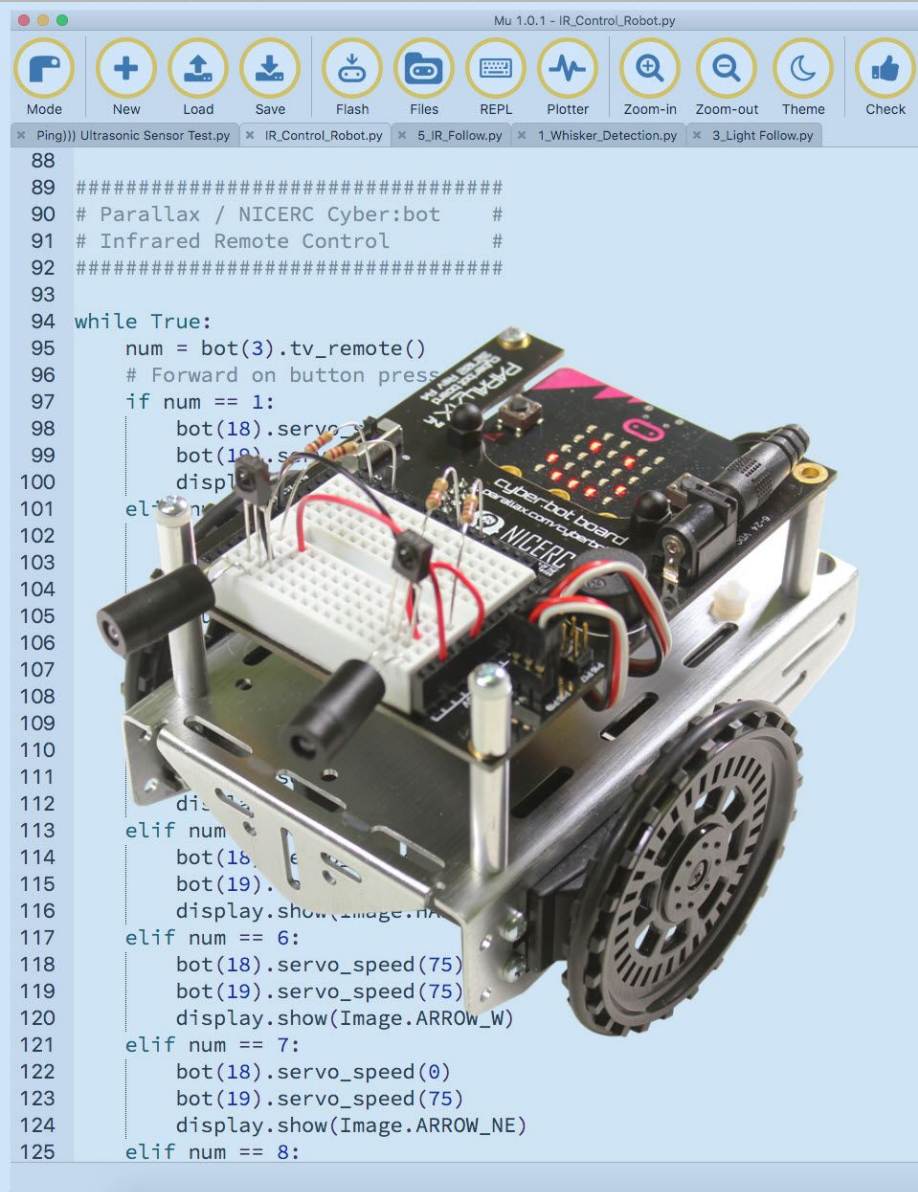
**NICERC**  
AN ACADEMIC DIVISION OF THE  
CYBER INNOVATION CENTER



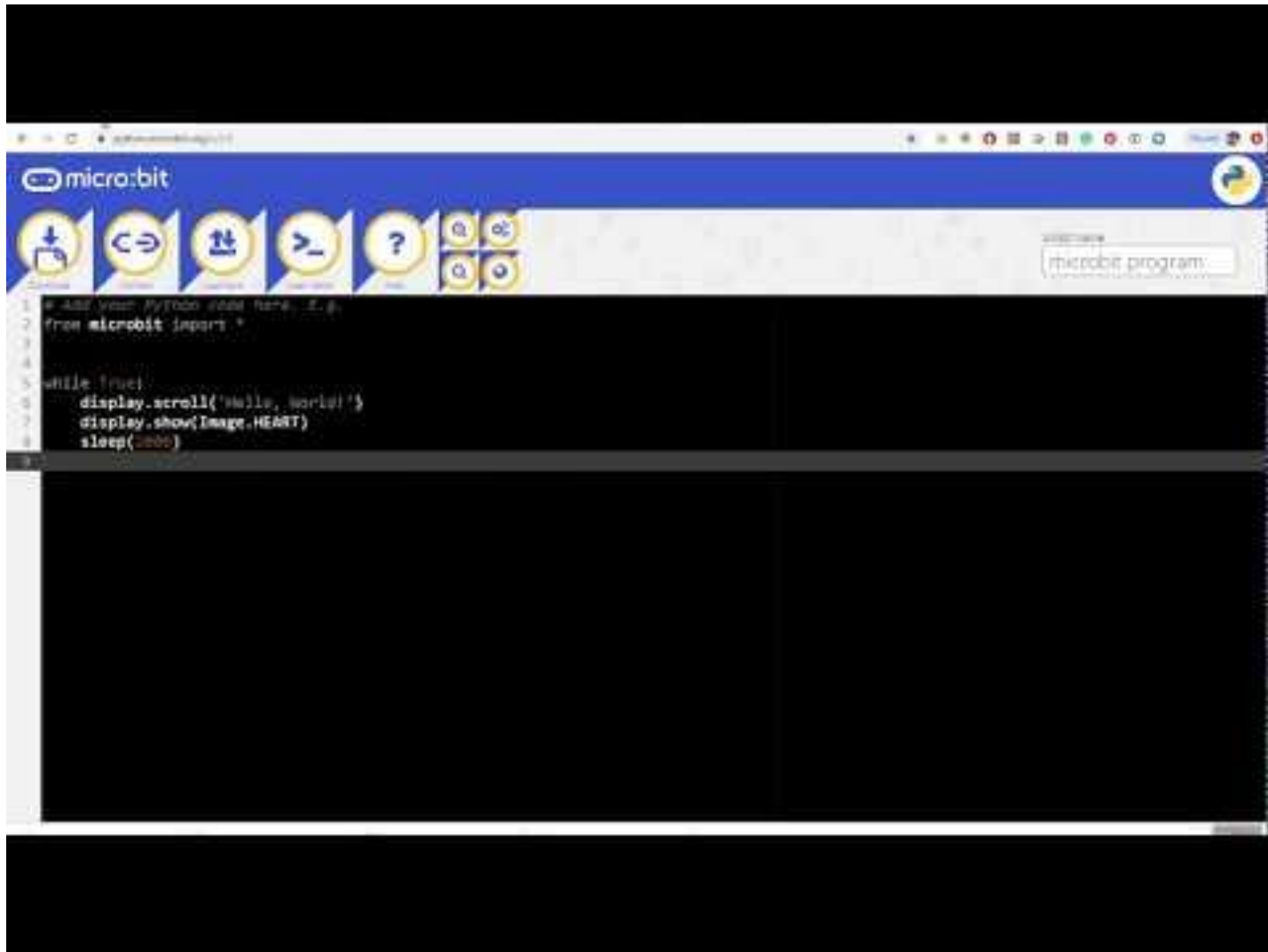


# Python + Robotics = cyber:bot

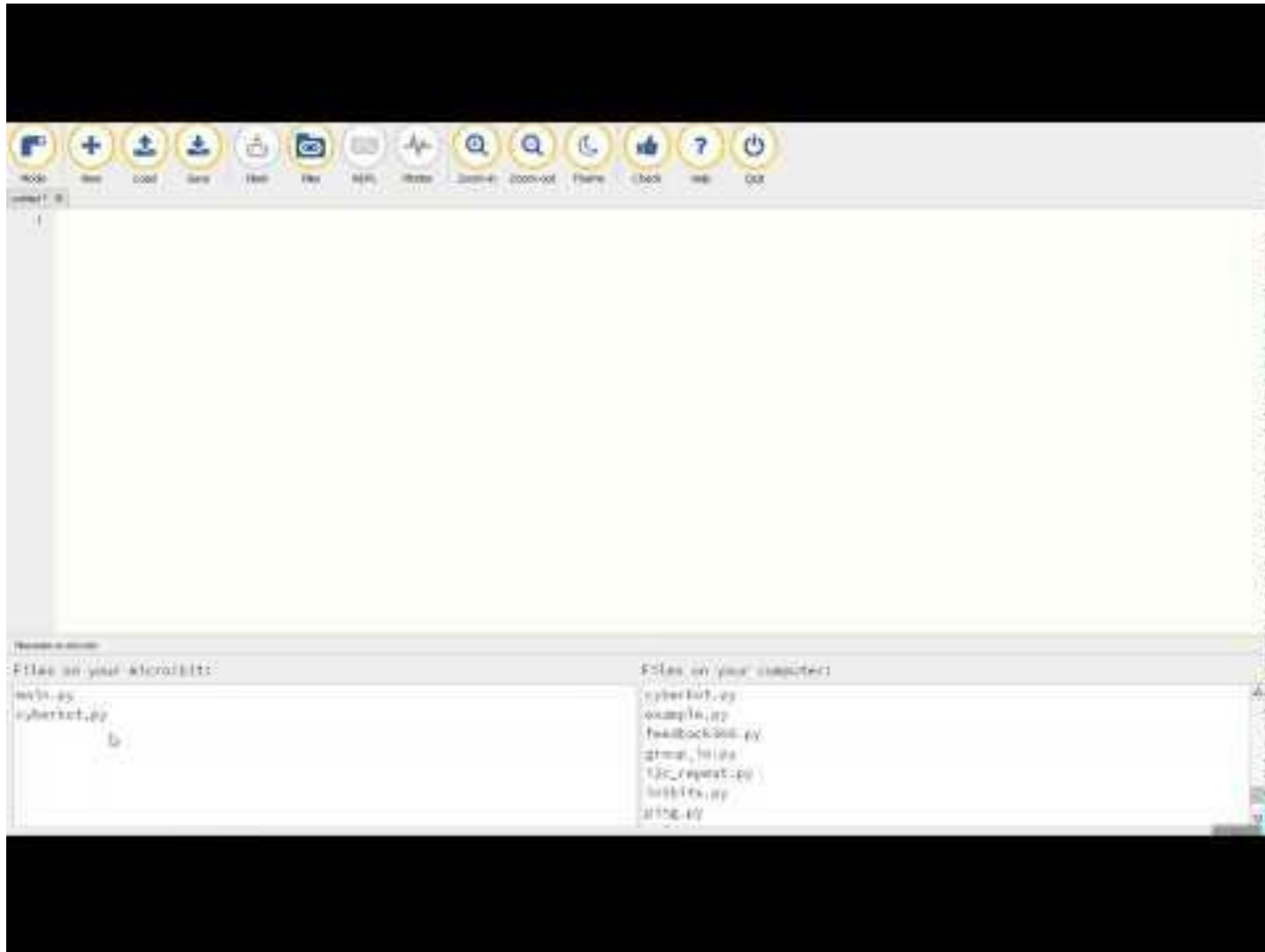
- Students learn more when they can “see” their programs run
- Competition-based challenges make it fun
- Basis for learning product development, robotics, and mechanical
- Understanding about “how it works” vs. consumer-level experience

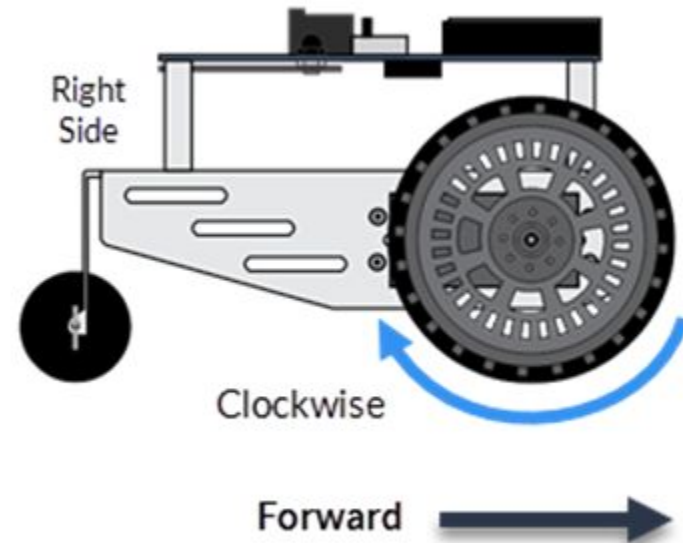
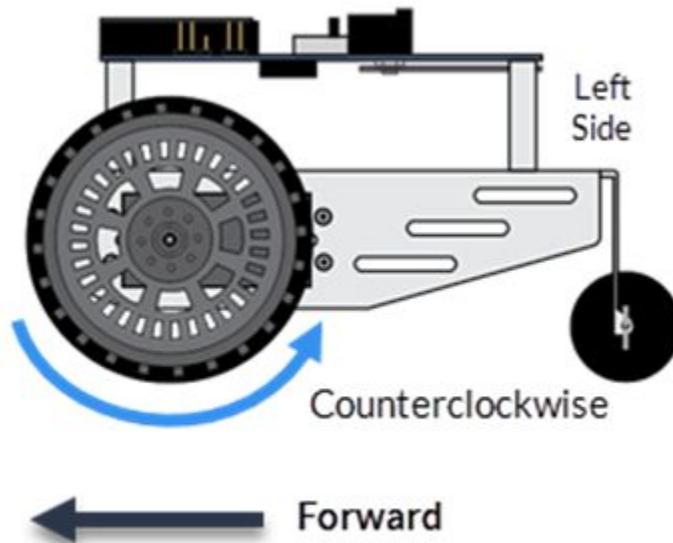


# Add cyberbot.py module to Microbit using **Python editor**

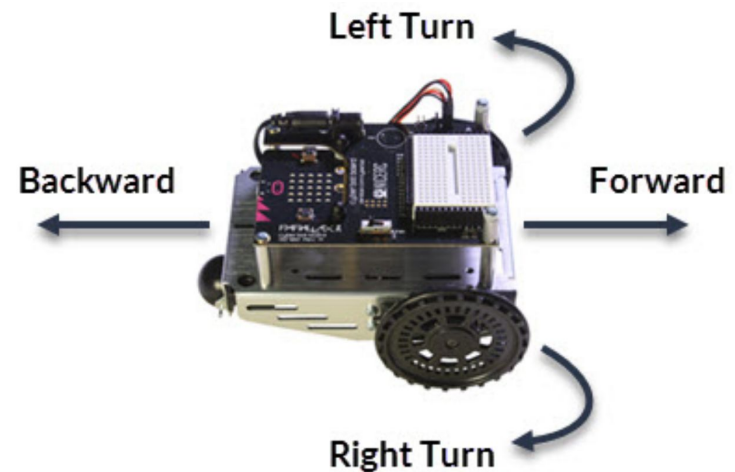


# Add cyberbot.py module to Microbit using Mu





**Forward Motion**  
Opposite directions  
for forward motion



**NICERC**  
AN ACADEMIC DIVISION OF THE  
CYBER INNOVATION CENTER

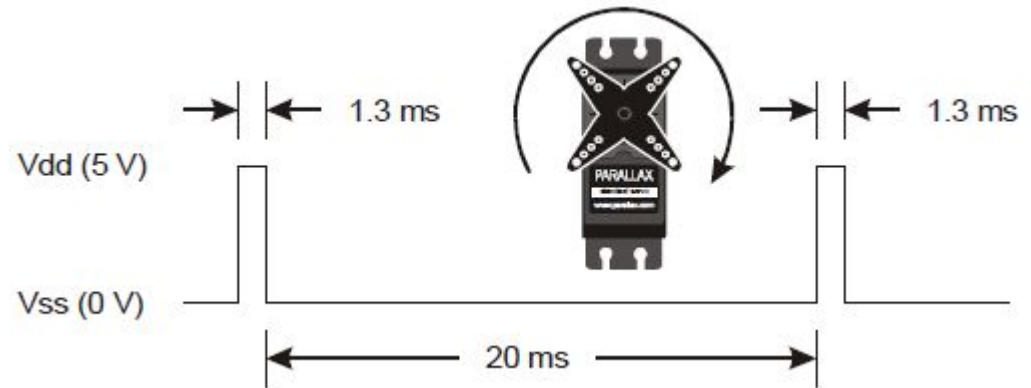




# Clockwise Rotation

```
# left_servo_CW.py
from cyberbot import *

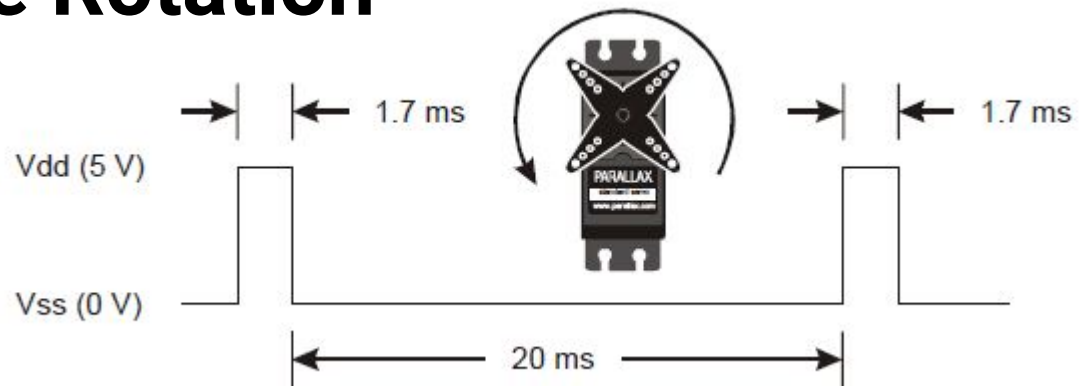
bot(18).servo_speed(-75)
```



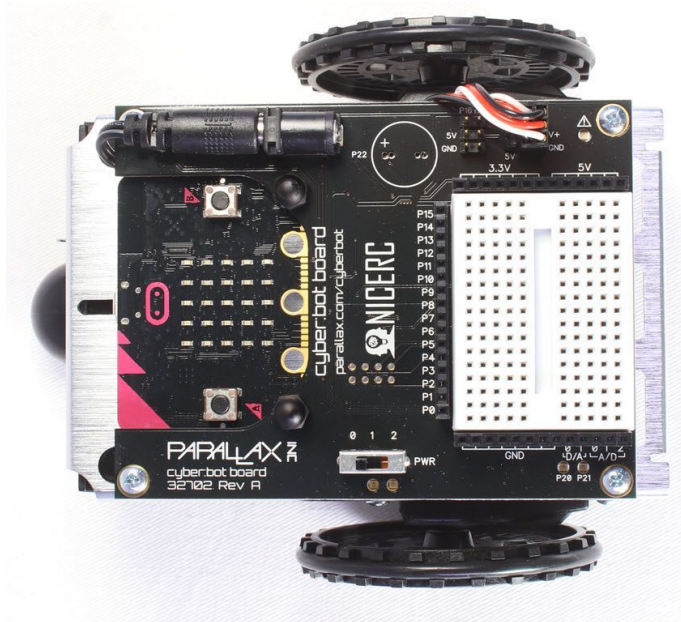
# Counter-clockwise Rotation

```
# left_servo_CCW.py
from cyberbot import *

bot(18).servo_speed(75)
```

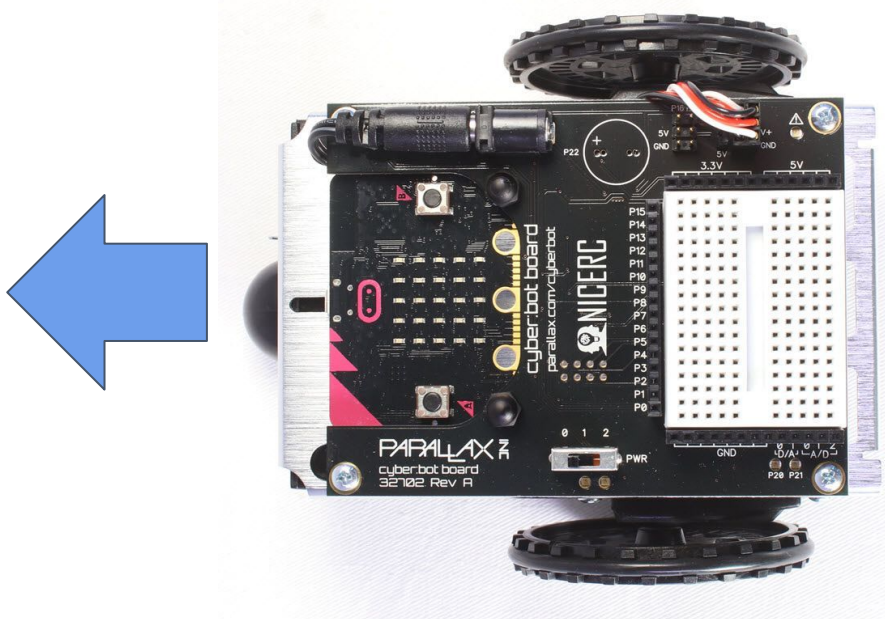


# Forward



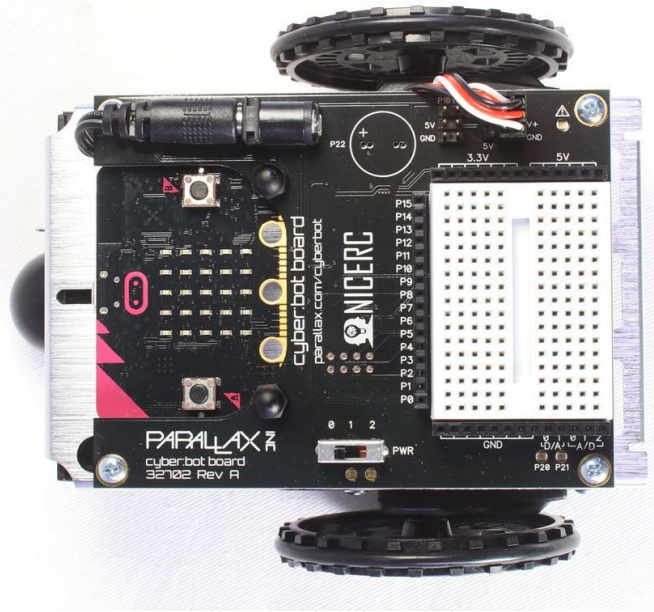
```
from cyberbot import *  
  
# forward_three_seconds.py  
  
bot(18).servo_speed(75)  
bot(19).servo_speed(-75)  
sleep (3000)  
  
# stop  
bot(18).servo_speed(0)  
bot(19).servo_speed(0)
```

# Backward



```
from cyberbot import *  
  
# backward_three_seconds.py  
  
bot(18).servo_speed(-75)  
bot(19).servo_speed(75)  
sleep (3000)  
  
# stop  
bot(18).servo_speed(0)  
bot(19).servo_speed(0)
```

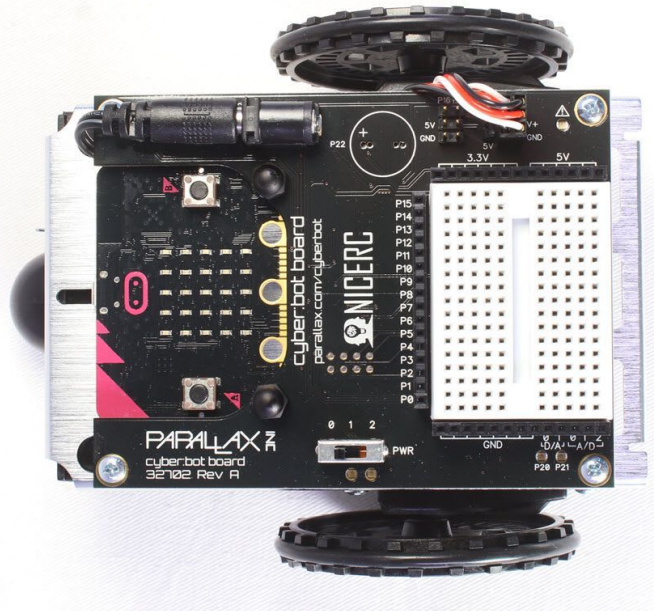
# Right



```
from cyberbot import *  
  
# right_three_seconds.py  
  
bot(18).servo_speed(75)  
bot(19).servo_speed(0)  
sleep (3000)  
  
# stop  
bot(18).servo_speed(0)  
bot(19).servo_speed(0)
```

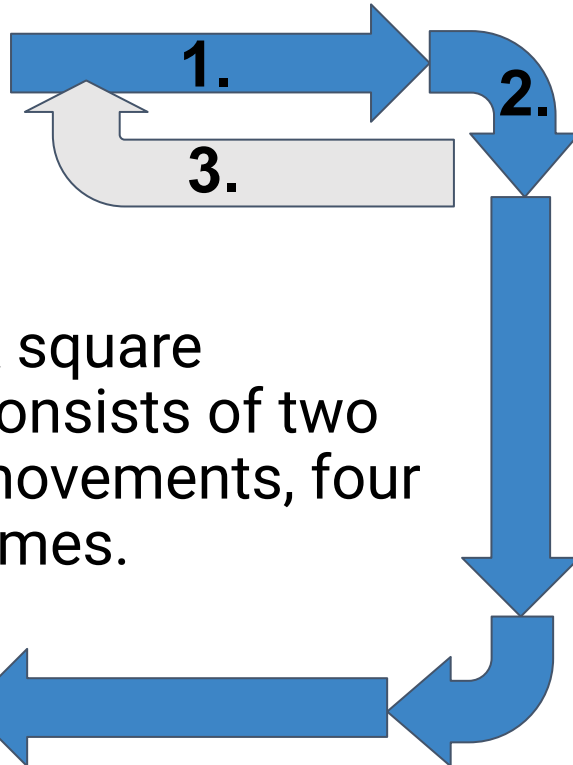
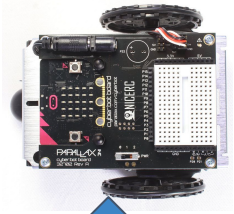


# Left



```
from cyberbot import *  
  
# left_three_seconds.py  
  
bot(18).servo_speed(0)  
bot(19).servo_speed(-75)  
sleep (3000)  
  
# stop  
bot(18).servo_speed(0)  
bot(19).servo_speed(0)
```

# Repeat Loops



A square  
consists of two  
movements, four  
times.

```
from cyberbot import *  
# square_with_repeat
```

```
for y in range (0, 3):  
    # straight  
    bot(18).servo_speed(75)  
    bot(19).servo_speed(-75)  
    sleep (3000)
```

```
    # right  
    bot(18).servo_speed(75)  
    bot(19).servo_speed(0)  
    sleep (2000)
```



**NICERC**  
AN ACADEMIC DIVISION OF THE  
CYBER INNOVATION CENTER



# Functions Without Arguments

- Simplified the drive commands
- Why not pass the duration?

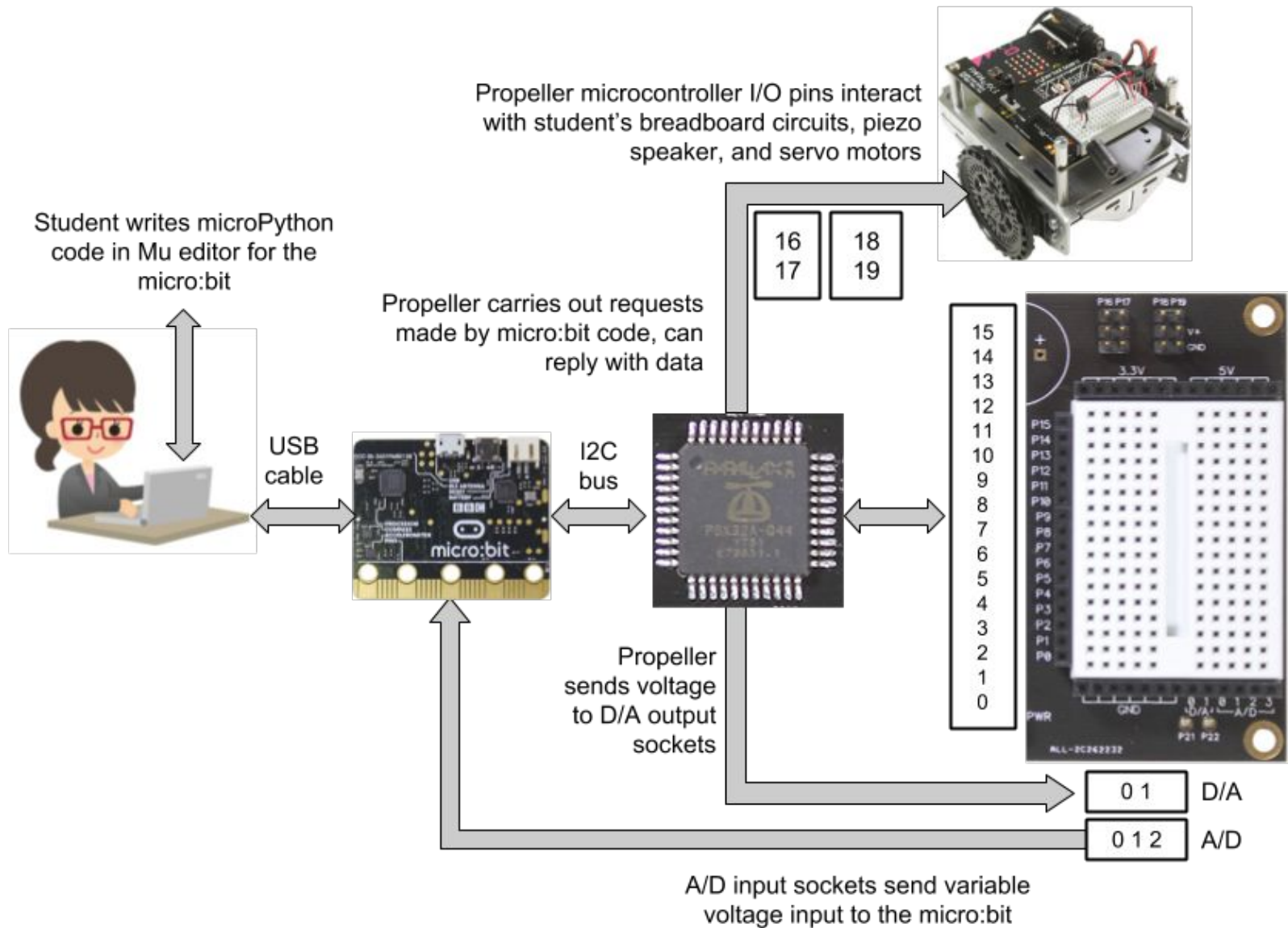
```
from cyberbot import *  
# functions with arguments  
  
def straight():  
    bot(18).servo_speed(75)  
    bot(19).servo_speed(-75)  
    sleep (3000)  
  
def right():  
    bot(18).servo_speed(75)  
    bot(19).servo_speed(0)  
    sleep (1100)  
  
def stop():  
    bot(18).servo_speed(0)  
    bot(19).servo_speed(0)  
  
straight()  
right()  
straight()  
right()  
stop()
```

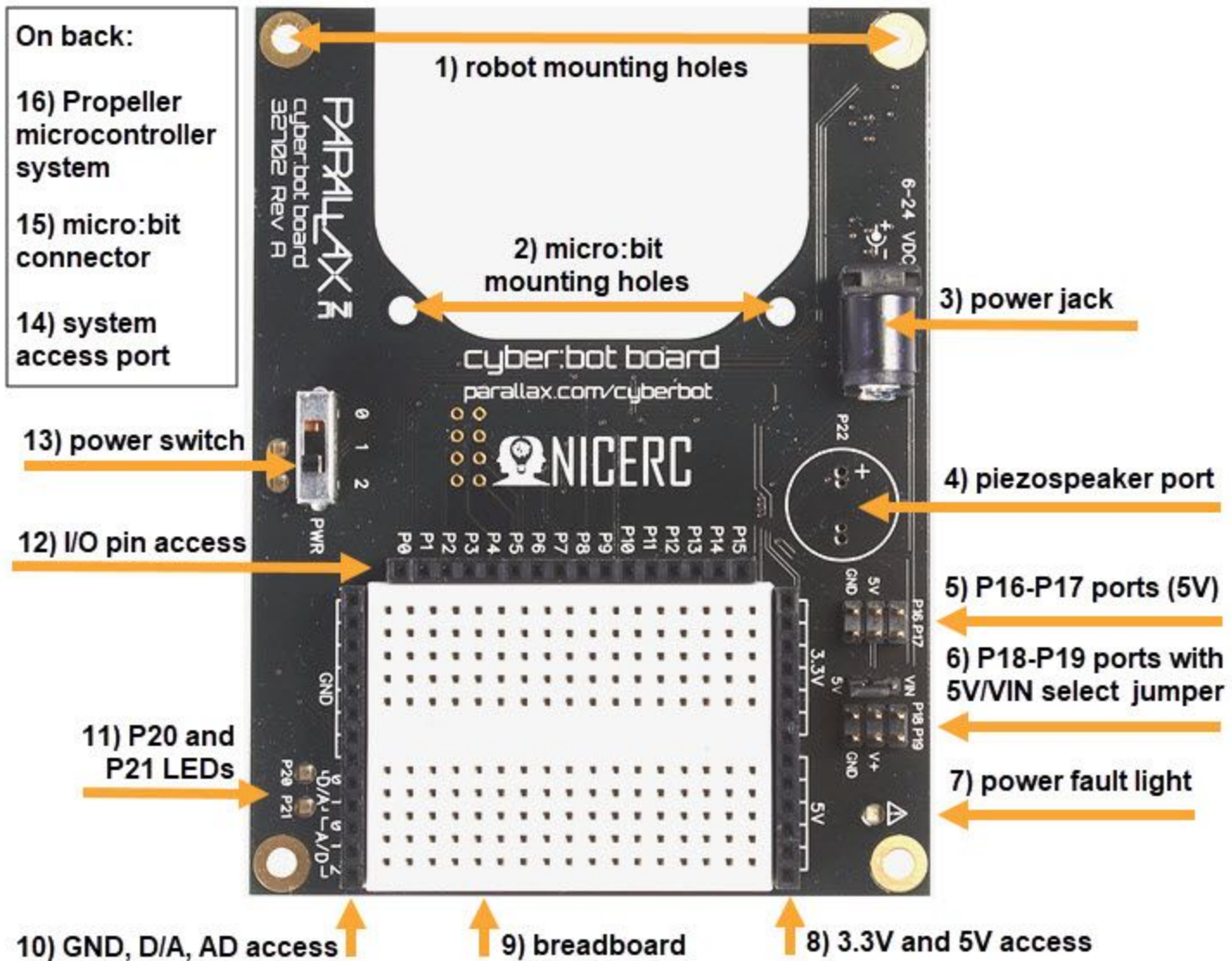


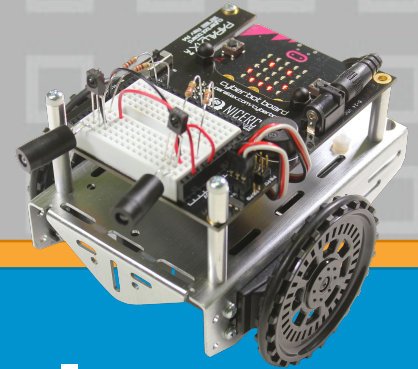
# Cyberbot.py library contents

<code>bot(pin).read_r(data)</code>	<code># retrieve returned value via I2C</code>
<code>bot(pin).digital_write(state)</code>	<code># set I/O pins high or low</code>
<code>bot(pin).analog_write(PWM)</code>	<code># set duty cycle to four available PWM channels</code>
<code>bot(pin).digital_read(state)</code>	<code># get I/O pin state high or low</code>
<code>bot(pin).states(states)</code>	<code># set binary pin states to multiple I/Os</code>
<code>bot(pin).directions(directions)</code>	<code># set I/O pin directions</code>
<code>bot(pin).qti(QTI values)</code>	<code># set and read four line follower sensors</code>
<code>bot(pin).pulse_out(pulsewidth)</code>	<code># set and maintain a pulse</code>
<code>bot(pin).pulse_in(pulsewidth)</code>	<code># measure pulse on I/O pin (accelerometers)</code>
<code>bot(pin).pulse_count(counts)</code>	<code># count pulses over duration of time</code>
<code>bot(pin).rc_time(time)</code>	<code># pseudo-analog R/C charge/discharge time on I/O pin</code>
<code>bot(pin).tone(sound)</code>	<code># set frequency, duration to I/O pin</code>
<code>bot(pin).ir_detect(frequency)</code>	<code># generate IR pulse and get receiver value</code>
<code>bot(pin).servo_angle(angle)</code>	<code># set and hold servo in an angle (up to 14 servos)</code>
<code>bot(pin).servo_speed(speed)</code>	<code># set and hold servo speed (-100 to 100)</code>
<code>bot(pin).servo_disable(disable)</code>	<code># disable a servo</code>
<code>bot(pin).ping_distance(distance)</code>	<code># configure Ultrasonic or Laser Ping, receive distance</code>
<code>bot(pin).tv_remote(button)</code>	<code># decode pulses from Sony TV remote and return button number</code>









# Cyber:bot Educational Resources

# Learn.parallax.com



LEARN.PARALLAX.COM

Search

WELCOME

TUTORIALS

EDUCATORS

REFERENCE

DOWNLOADS

Are you an Educator? Let's chat!



cyber:bot

Tutorial Series



## Get a Head Start on Python Robotics with the cyber:bot Robot

Take Python programming to the next level with the new cyber:bot Robot! This brand-new robot is a joint project of Parallax and NICERC, and our tutorial series is in active development.

Come take a look at its progress, and remember to check back often over the next few months.

## Grade

### Grades 5-8

ActivityBot Boe-Bot Scribbler 3 FLiP Try-It Kit Shield-Bot

### Grades 9+

ActivityBot cyber:bot Boe-Bot Scribbler 3 FLiP Try-It Kit Shield-Bot ELEV-8 Arlo

## Language

### BlocklyProp

Basics ActivityBot Scribbler 3 FLiP Try-It Kit Badge WX

### Propeller C

Prop C Basics ActivityBot FLiP Try-It Kit



NICERC  
AN ACADEMIC DIVISION OF THE  
CYBER INNOVATION CENTER





# Assemble your cyber:bot

<https://learn.parallax.com>



(2) 7/8" #4-40 pan-head screw  
#710-00007



(10) 3/8" #4-40 pan-head screw  
#700-00002



(10) 1/4" #4-40 pan-head screw  
#700-00028



(2) 3/8" #4-40 Nylon flat-head screw  
#710-00046



(1) cotter pin  
#700-00023



(4) 1" round #4-40 standoff  
#700-00040



(2) 1/2" round #4 spacer  
#713-00007



(1) rubber grommet  
#700-00025



(2) #4 Nylon washer  
#700-00005



(8) #4-40 Nylon-core locknut  
#700-00024

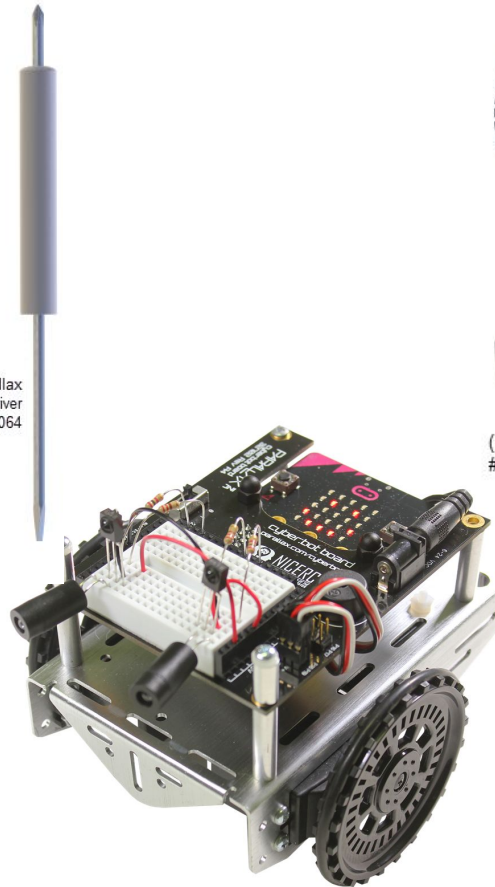


(8) #4-40 nut  
#700-00003



Parallax combination wrench  
#700-10025

Parallax  
Screwdriver  
#700-00064



(2) wheels  
#721-00021



(1) small robot chassis  
#700-00022



(2) o-ring tires  
#710-00200



(1) 5 AA battery holder  
#753-00007



(1) tail wheel ball  
#700-00099



(2) continuous  
rotation servos  
#900-00008

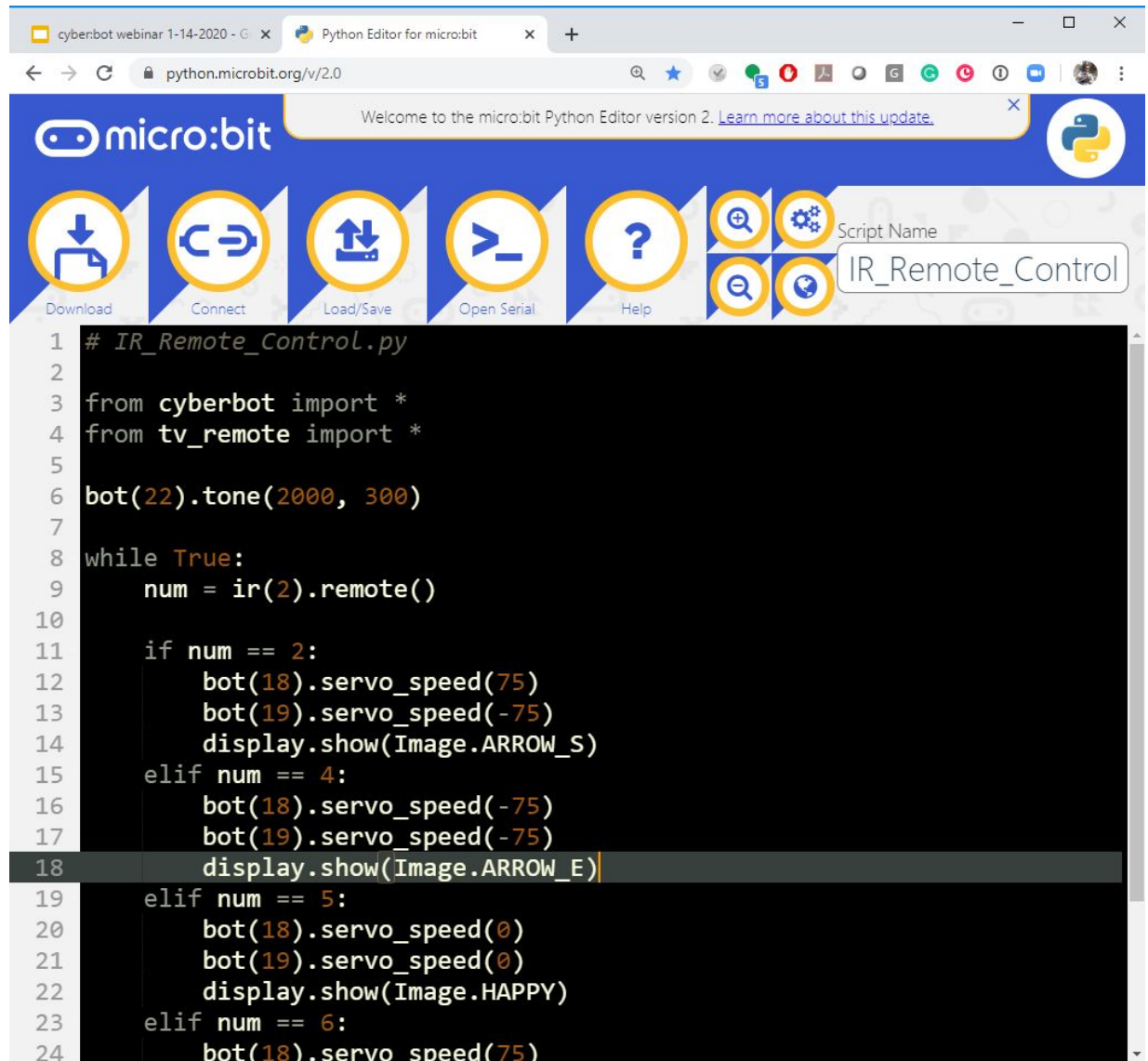


**NICERC**  
AN ACADEMIC DIVISION OF THE  
CYBER INNOVATION CENTER



# Python scripting: learn\* data types, logic, interaction between hardware and software

\*student experience,  
not instructed!

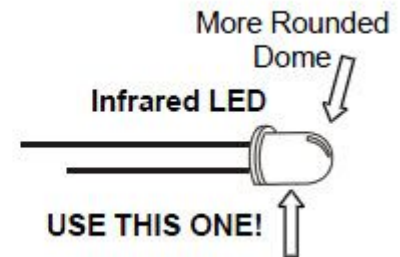
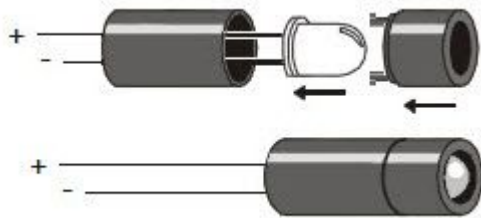


The screenshot shows the micro:bit Python Editor version 2.0 interface. The browser address bar displays 'python.microbit.org/v/2.0'. The editor's title bar includes 'cyberbot webinar 1-14-2020 - G' and 'Python Editor for micro:bit'. The main interface features a blue header with the 'micro:bit' logo and a 'Welcome to the micro:bit Python Editor version 2.0' message. Below the header is a toolbar with icons for Download, Connect, Load/Save, Open Serial, Help, and a grid of icons for Search, Settings, and other functions. A 'Script Name' input field contains the text 'IR\_Remote\_Control'. The code editor area displays a Python script for controlling a micro:bit robot using an IR remote. The script is as follows:

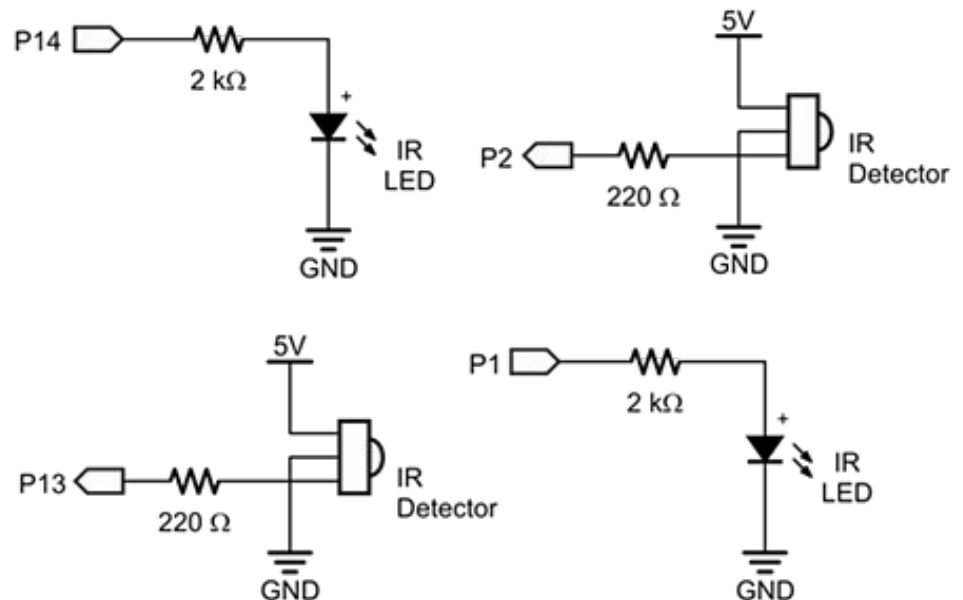
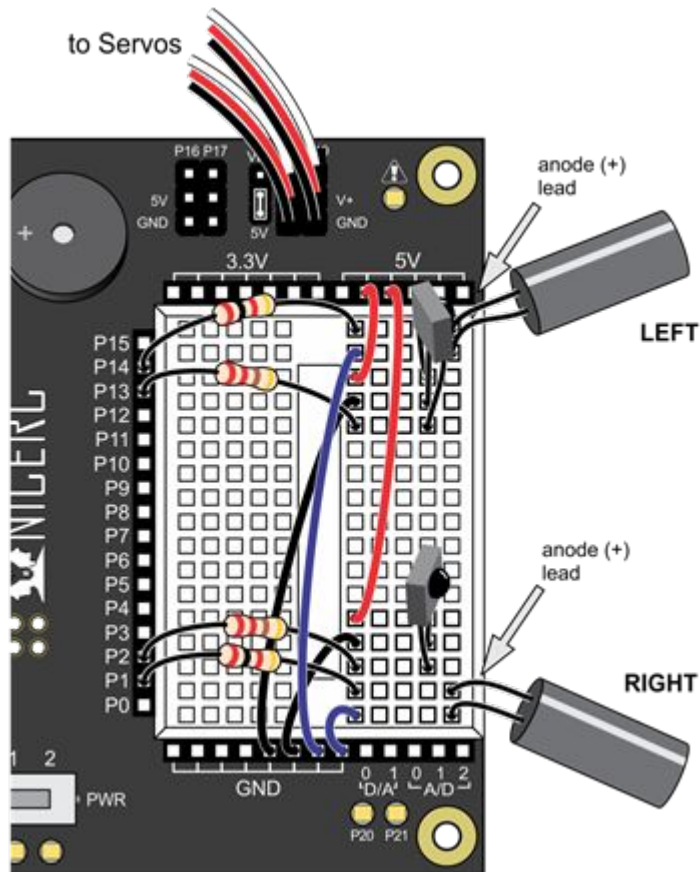
```
1 # IR_Remote_Control.py
2
3 from cyberbot import *
4 from tv_remote import *
5
6 bot(22).tone(2000, 300)
7
8 while True:
9     num = ir(2).remote()
10
11     if num == 2:
12         bot(18).servo_speed(75)
13         bot(19).servo_speed(-75)
14         display.show(Image.ARROW_S)
15     elif num == 4:
16         bot(18).servo_speed(-75)
17         bot(19).servo_speed(-75)
18         display.show(Image.ARROW_E)
19     elif num == 5:
20         bot(18).servo_speed(0)
21         bot(19).servo_speed(0)
22         display.show(Image.HAPPY)
23     elif num == 6:
24         bot(18).servo_speed(75)
```

# Components: identification, function and use of real electronic sensors and parts

- (2) IR receivers
- (2) IR LEDs (clear case)
- (2) IR LED shield assemblies
- (2) Resistors, 220  $\Omega$  (red-red-brown)
- (2) Resistors, 2 k $\Omega$  (red-black-red)
- (misc) Jumper wires



# Electronic circuit building by pictorial and schematic





# **Content exclusively available to educators:** e-mail [learn@parallax.com](mailto:learn@parallax.com) for access

- Google Presentation Slides
- Scope and sequence
- Standards alignment
- Assessment material

# Standards Alignment:

## Common Core (ELA, Math), NGSS, K-12 Computer Science Framework, CTE and 21st Century Competencies

Concepts Vocabulary	Common Core State Standards (ELA) <sup>2</sup>	Common Core State Standards (Math) <sup>2</sup>	Next Generation Science Standards (NGSS) <sup>4</sup>	K-12 Computer Science Framework <sup>1</sup>	Career Technical Education Standards (CTE) <sup>3</sup>	21st Century Competencies
<ul style="list-style-type: none"> <li>• Active-low</li> <li>• Active-high</li> <li>• Anode</li> <li>• Cathode</li> <li>• Circuit</li> <li>• Diode</li> <li>• Jumper wire</li> <li>• LED</li> <li>• Ohms</li> <li>• Prototyping</li> <li>• Pushbutton</li> <li>• Pull-down resistor</li> <li>• Pull-up resistor</li> <li>• Resistor</li> <li>• Socket</li> </ul>	<p>CCSS.ELA-LITERACY.CCRA.R.1 Read closely to determine what the text says explicitly and to make logical inferences from it; cite specific textual evidence when writing or speaking to support conclusions drawn from the text.</p> <p>CCSS.ELA-LITERACY.CCRA.R.4 Interpret words and phrases as they are used in a text, including determining technical, connotative, and figurative meanings, and analyze how specific word choices shape meaning or tone.</p> <p>CCSS.ELA-LITERACY.CCRA.R.7 Integrate and evaluate content presented in diverse media and formats, including visually and quantitatively, as well as in words.</p> <p>CCSS.ELA-LITERACY.CCRA.R.10 Read and comprehend complex literary and informational texts independently and proficiently.</p>			<p><i>Practices</i></p> <p>P4: Developing and Using Abstractions. 1 <i>Extract common features from a set of interrelated processes or complex phenomena.</i></p> <p>P4: Developing and Using Abstractions. 2 <i>Evaluate existing technological functionalities and incorporate them into new designs.</i></p> <p>P4: Developing and Using Abstractions. 3 <i>Create modules and develop points of interaction that can apply to multiple situations and reduce complexity.</i></p> <p>P5: Creating Computational Artifacts. 1 <i>Plan the development of a computational artifact using an iterative process that includes reflection on and modification of the plan, taking into account key features, time and resource constraints, and user expectations.</i></p> <p>P5: Creating Computational Artifacts. 2 <i>Create a computational artifact for practical intent, personal expression, or to address a societal issue.</i></p> <p>P5: Creating Computational Artifacts. 3 <i>Modify an existing artifact to improve or customize it.</i></p> <p>P6: Testing and Refining Computational Artifacts. 1 <i>Systematically test computational artifacts</i></p>	<p>(CRP) Career Ready Practices</p> <p>2. Apply appropriate academic skills</p> <p>11. Use technology to enhance productivity</p> <p>(ST) Stem Careers</p> <p>2. Use technology to acquire, manipulate, analyze and report data.</p> <p>6. Demonstrate technical skills needed in a chosen STEM field.</p> <p>(ST-ET) Engineering &amp; Technology</p> <p>1. Use STEM concepts and processes to solve problems involving design and/or production.</p> <p>3. Apply processes and concepts for the use of technological tools in STEM.</p> <p>5. Apply the elements of the design process.</p> <p>6. Apply the knowledge learned in STEM to solve problems.</p> <p>(ST-SM) Science &amp; Math</p> <p>2. Apply science and mathematics concepts to the development of plans.</p>	<p>Self-direction</p> <p>Technology Use</p> <p>Innovation</p> <p>Critical-thinking</p> <p>Reflection</p> <p>Revision</p> <p>Design-thinking</p>



# Assessment material:

editable (RTF) and PDF formats available for download

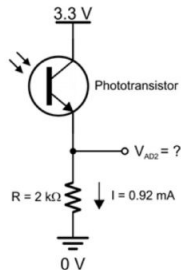
Python Programming with the cyber:bot

Visible Light Navigation

Name: \_\_\_\_\_ Class: \_\_\_\_\_

Date: \_\_\_\_\_

- 11.) Capacitor decay time is based on \_\_\_\_\_.  
a. resistor size c. brightness of light  
b. voltage d. the number of capacitors used
- 12.) When roaming with visible light, the key variable is the \_\_\_\_\_ the measurements of two sensors.  
a. sum of c. average of  
b. difference between d. median of
- 13.) Using the Propeller microcontroller, decay time is \_\_\_\_\_.  
a. the time it takes for the capacitor to register 0 V.  
b. the voltage value after 1 ms.  
c. the difference between the initial voltage and the ending voltage.  
d. the time it takes for the capacitor to reach 1.6 V.
- 14.) Use Ohm's Law to calculate the volts at AD2. \_\_\_\_\_



Answer true or false to questions 15 - 19.

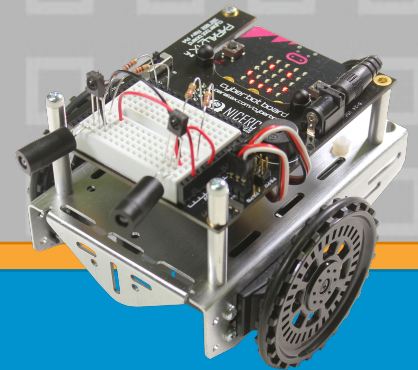
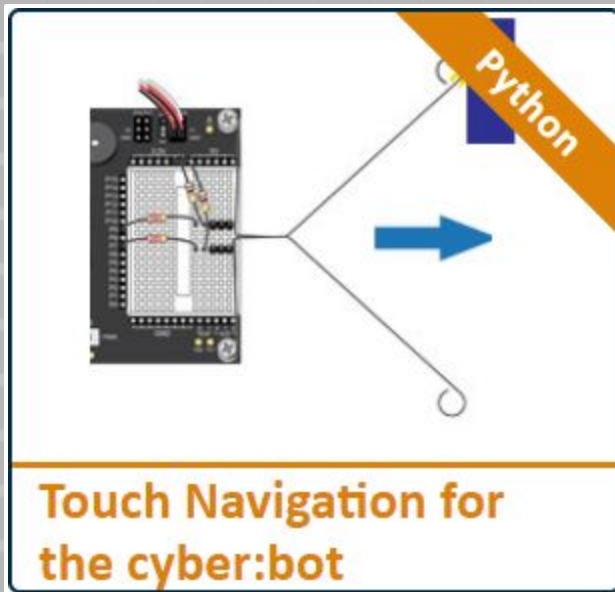
- 15.) \_\_\_\_\_ Brighter light results in less current.
- 16.) \_\_\_\_\_ Zero-justified normalized differential measurement centers the data around zero.
- 17.) \_\_\_\_\_ Current flows through the phototransistor in two directions.
- 18.) \_\_\_\_\_ A smaller resistor in series with a phototransistor makes the circuit less sensitive to light than when a larger resistor is used.
- 19.) \_\_\_\_\_ In bright light the capacitor drains more quickly resulting in a low reading.
- 20.) List 3 actions of the `rc_time` function.  
\_\_\_\_\_

- Developed from cyber:bot tutorial content
- One assessment per cyber:bot “chapter”
- Answer key is included
- Use as provided or modify on your own

# Scope and Sequence:

- [Google Drive - look for tab](#)
- [Download PDF & zipped Excel](#)

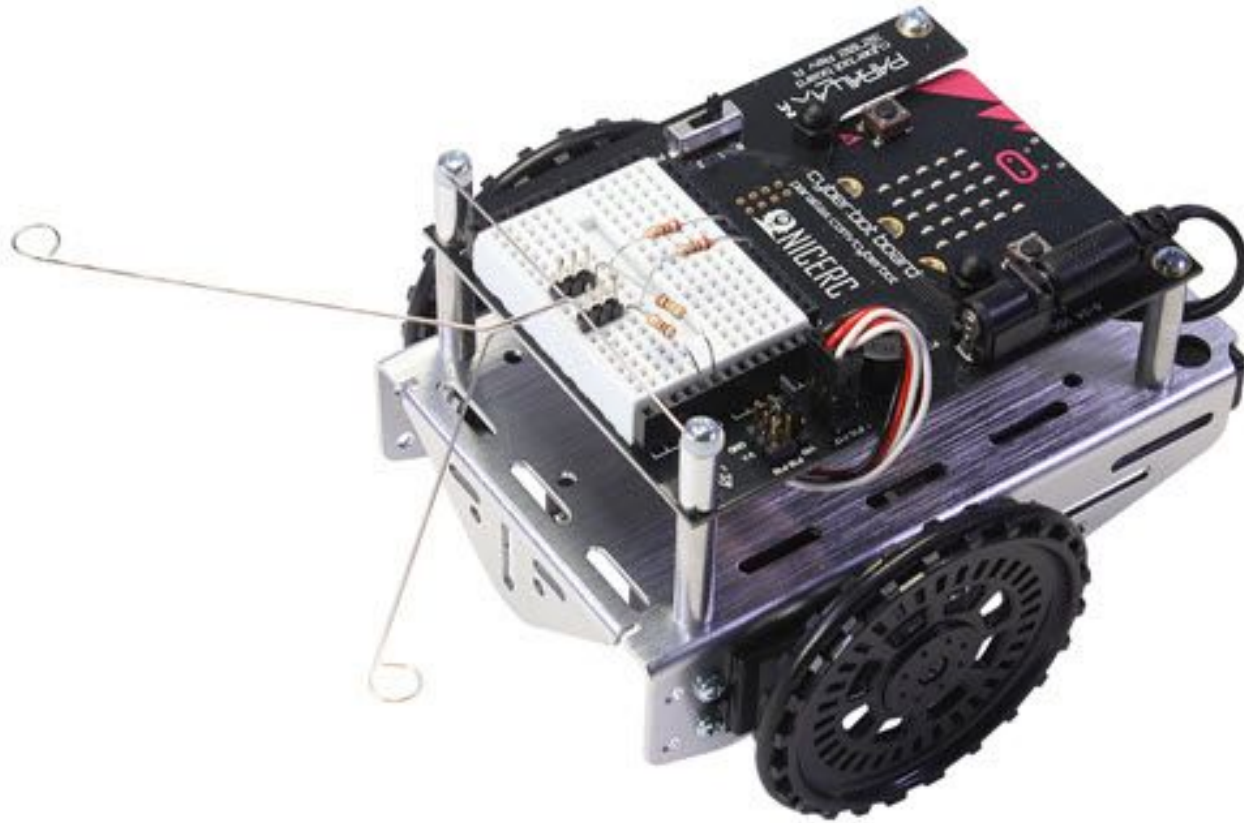




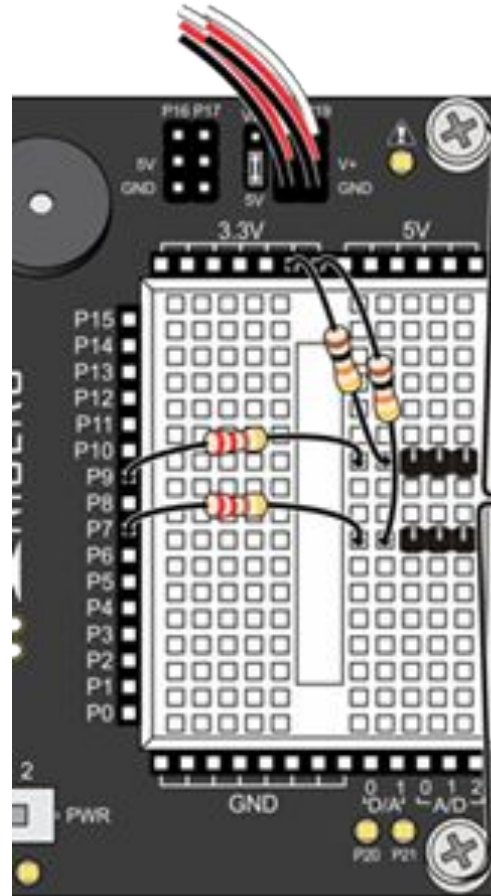
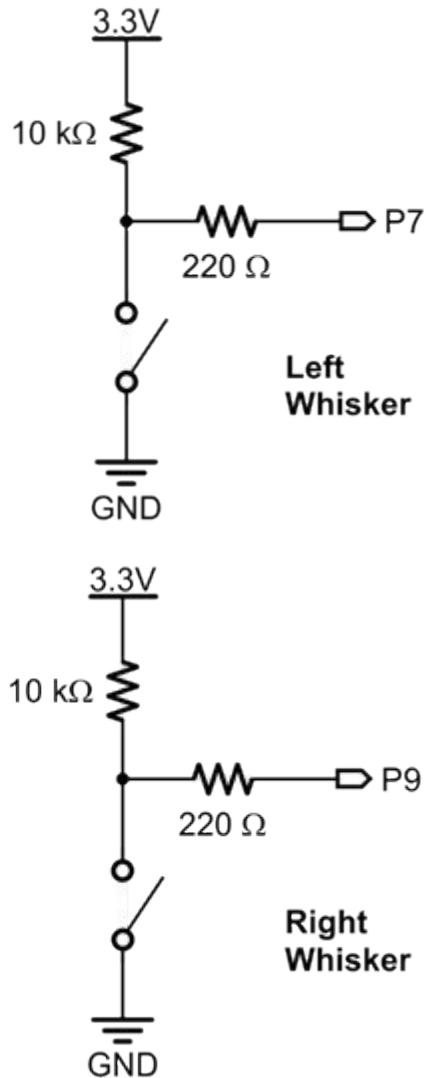
# Touch Navigation

<https://learn.parallax.com/tutorials/robot/cyberbot/touch-navigation-cyberbot>

# Touch Navigation: Assembled Circuit



# Touch Navigation: Circuit Build



Left  
Whisker

Right  
Whisker

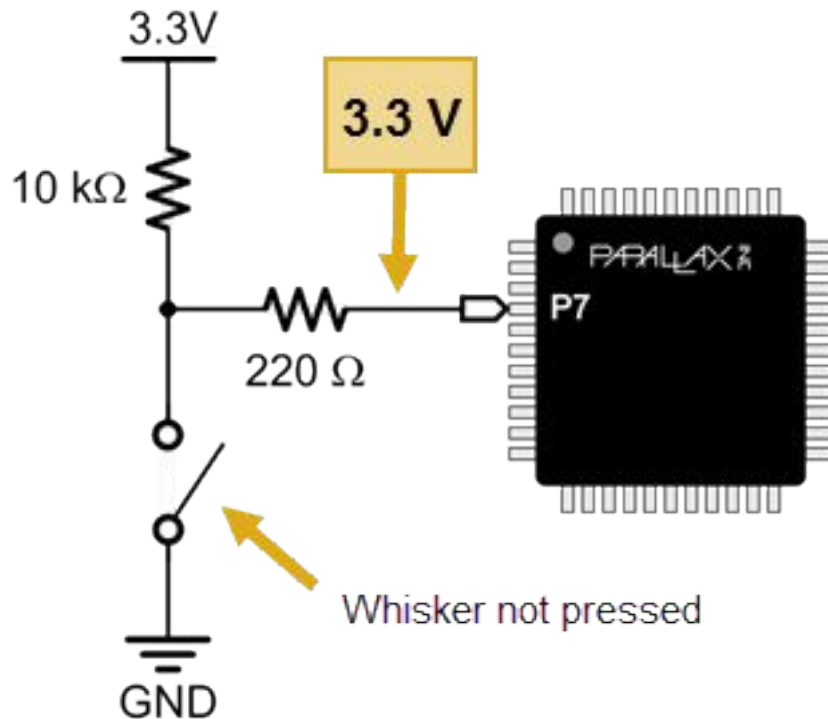


**NICERC**  
AN ACADEMIC DIVISION OF THE  
CYBER INNOVATION CENTER

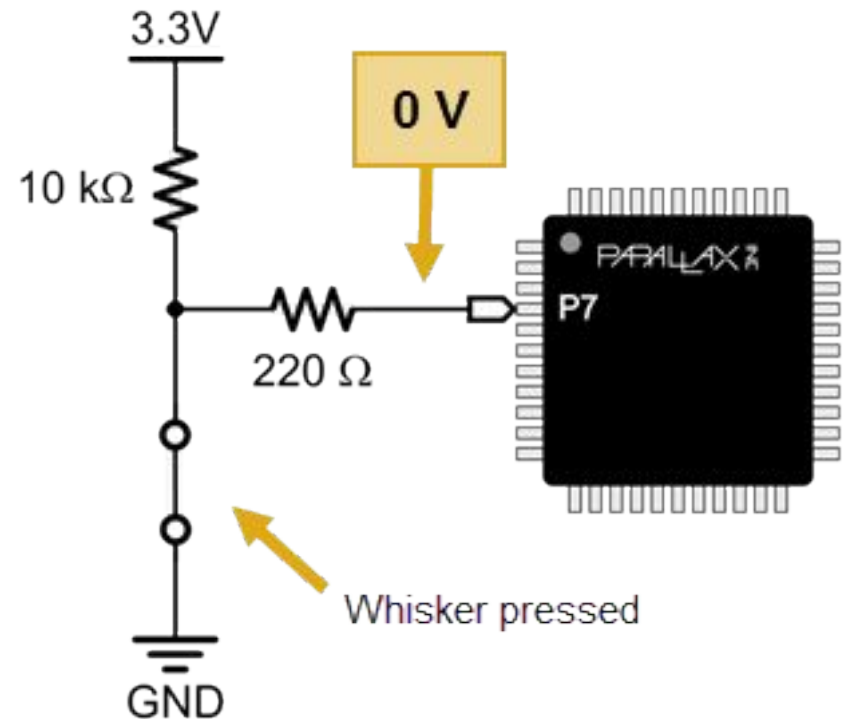


# Touch Navigation: Pressed / Not Pressed

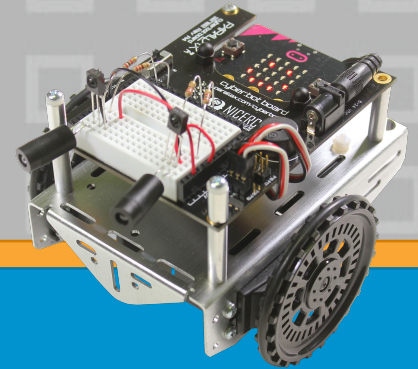
`bot(7).read_digital()`  
returns 1



`bot(7).read_digital()`  
returns 0



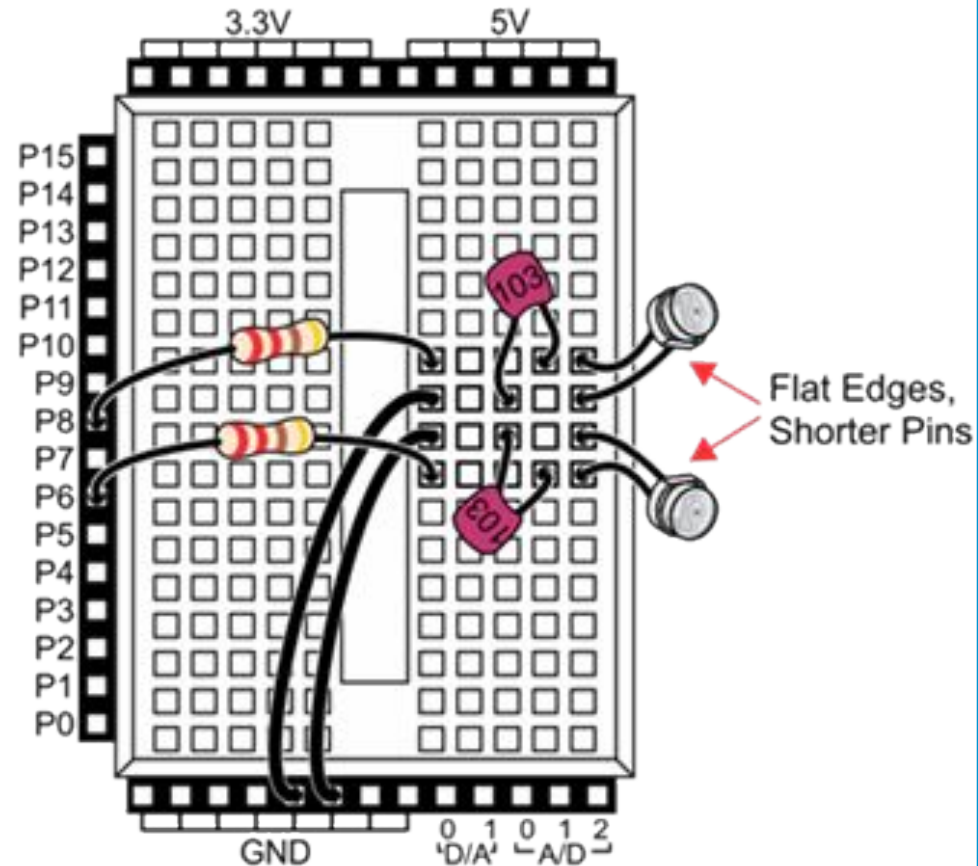
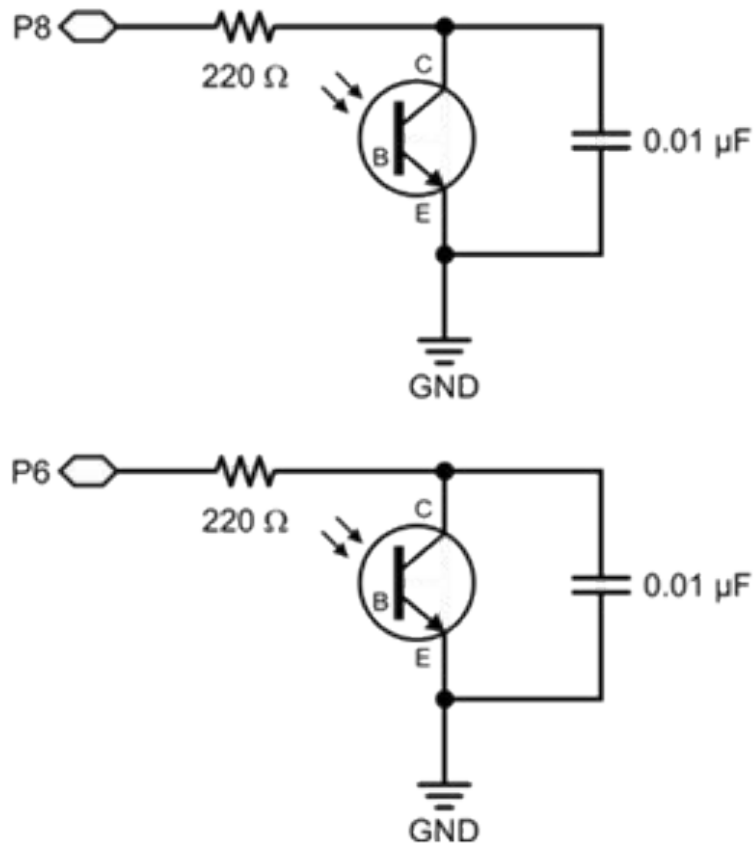




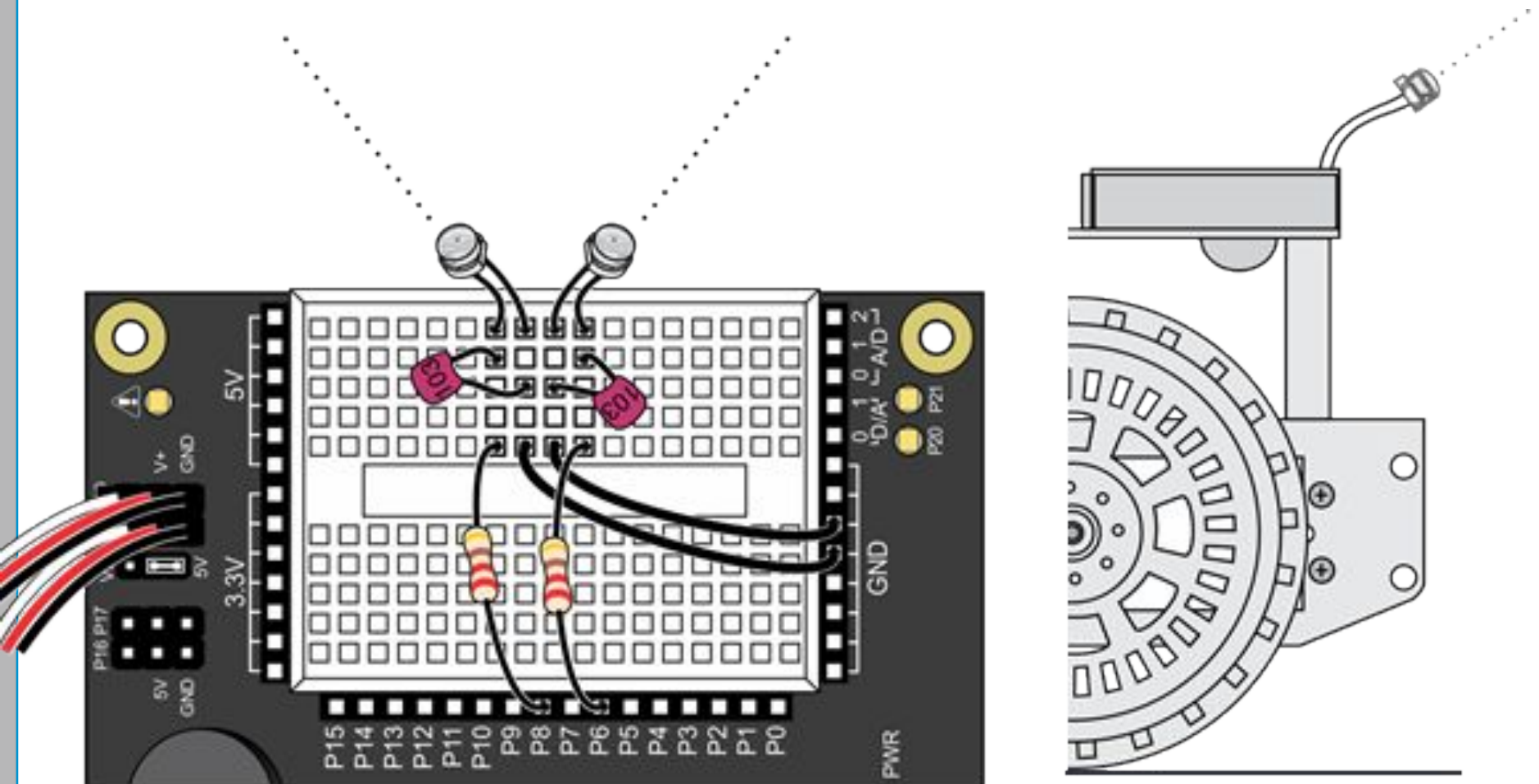
# Visible Light

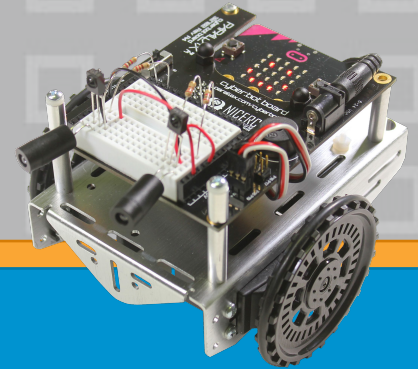
<https://learn.parallax.com/tutorials/robot/cyberbot/visible-light-navigation-cyberbot>

# Visible Light Following: Circuit Build



# Visible Light Following: Circuit Build



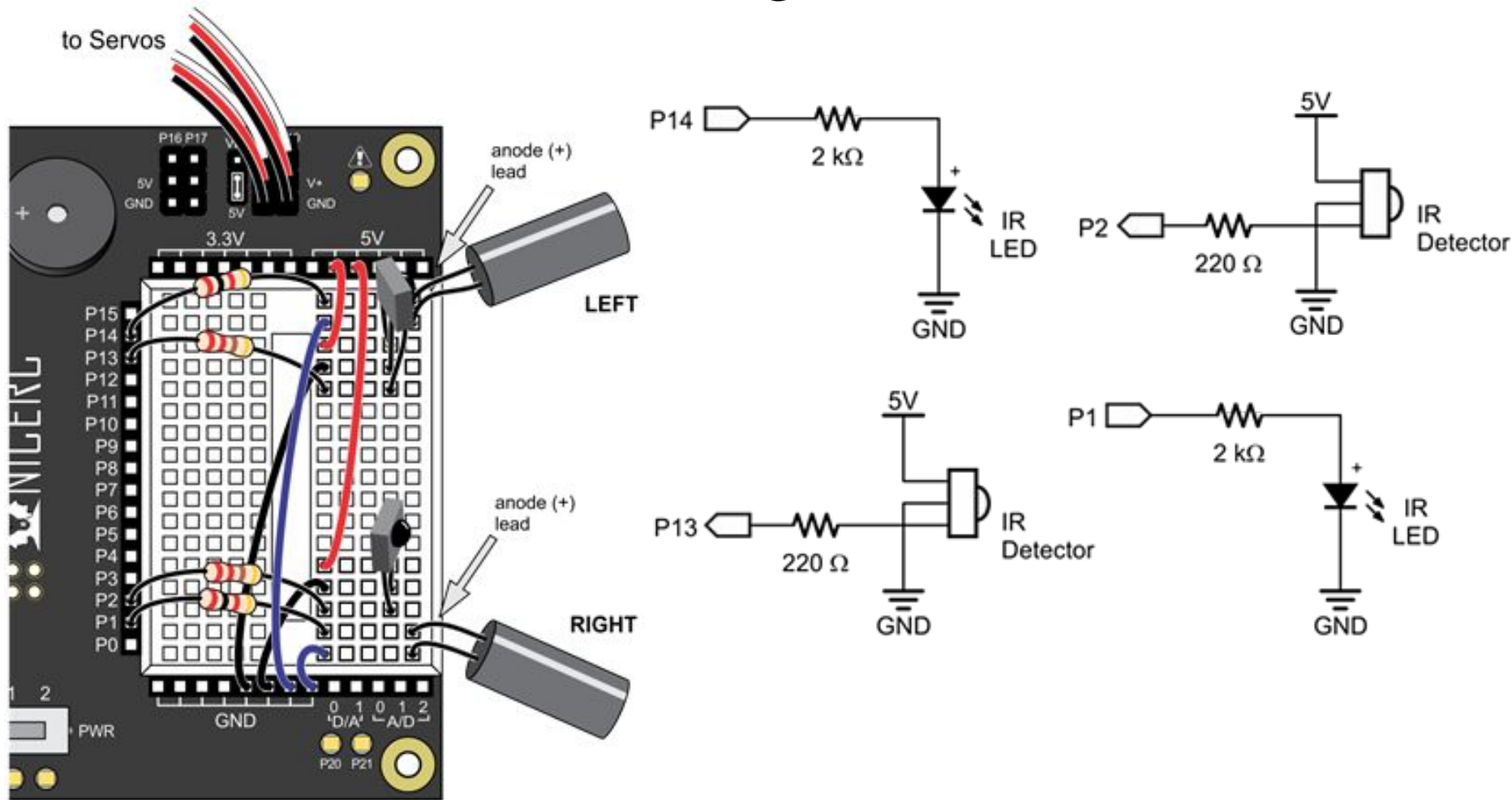


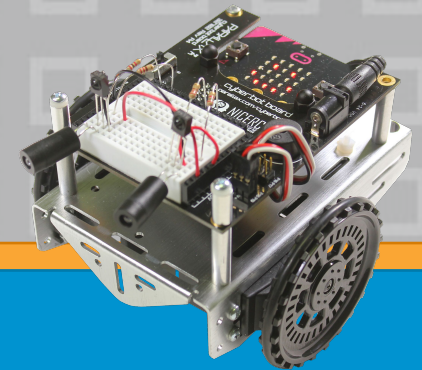
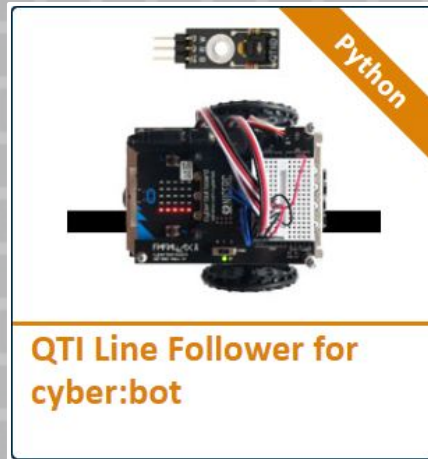
# Infrared Light

<https://learn.parallax.com/tutorials/robot/cyberbot/infrared-light-navigation-cyberbot>



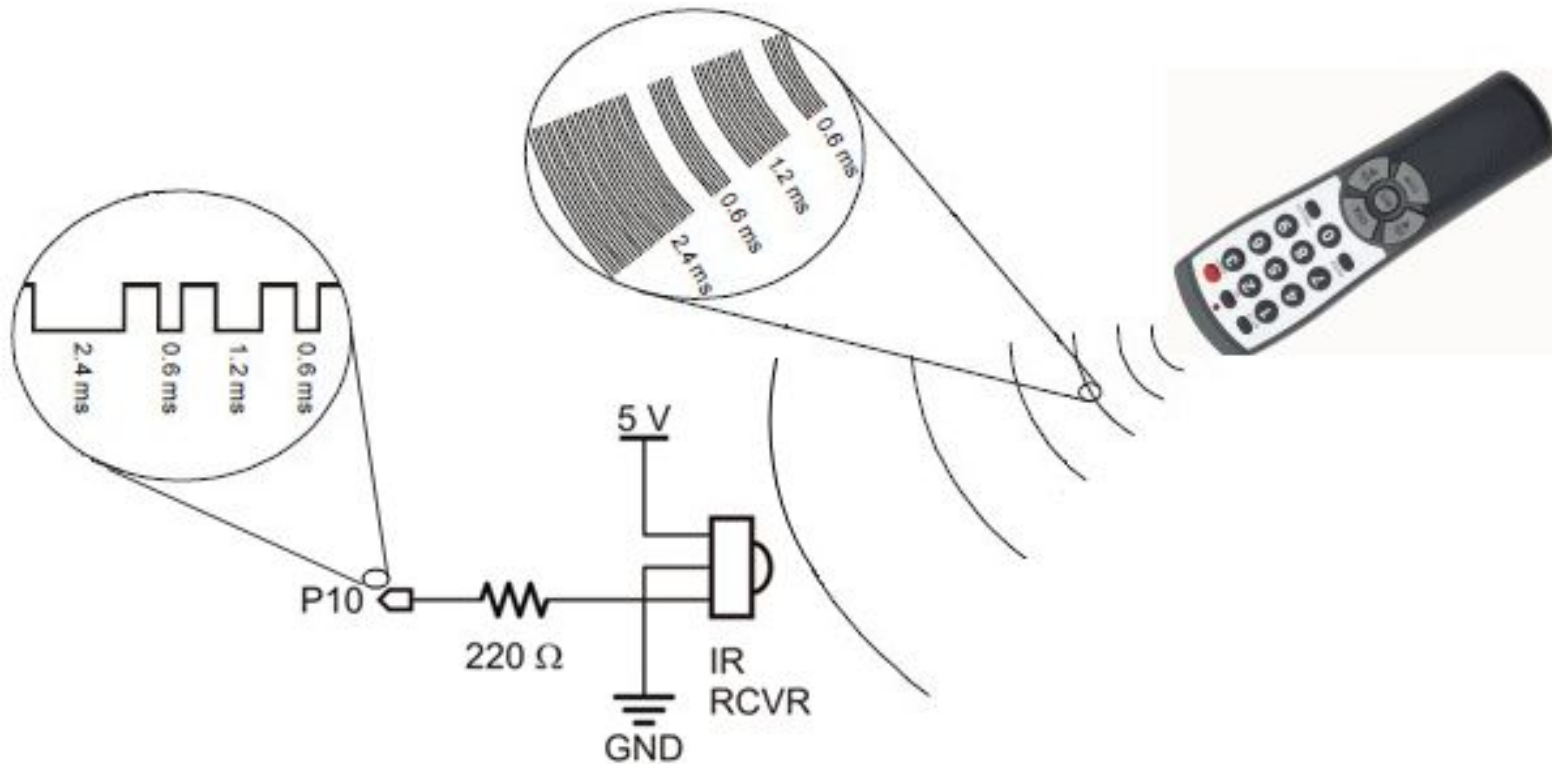
# Infrared Emitter / Receivers for Object Detection and Following





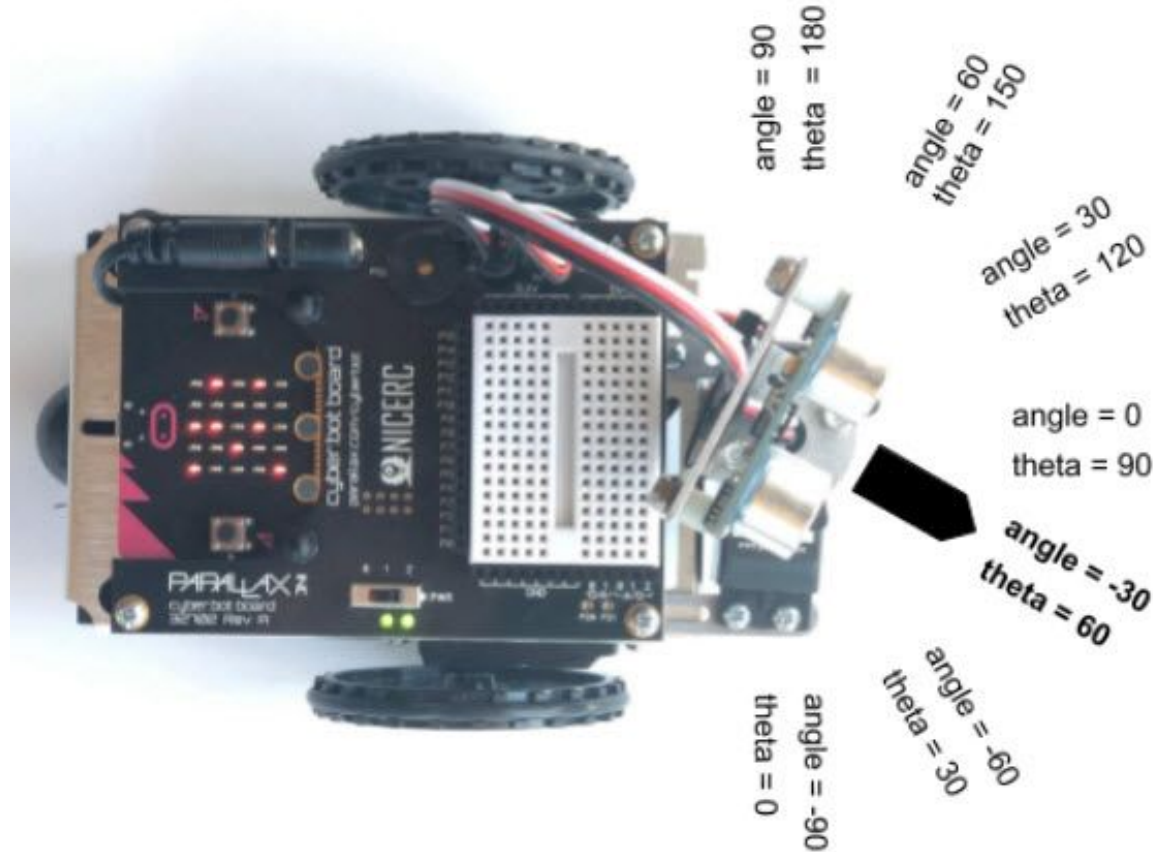
# Projects

# Infrared Remote Control



<http://learn.parallax.com/tutorials/robot/cyberbot/control-your-cyberbot-infrared-tv-remote>

# Roaming with the Ping))) Ultrasonic Sensor



<http://learn.parallax.com/tutorials/robot/cyberbot/cyberbot-roaming-ping>



# Happening Now: NICERC & Parallax Cybersecurity Curriculum Jam Session

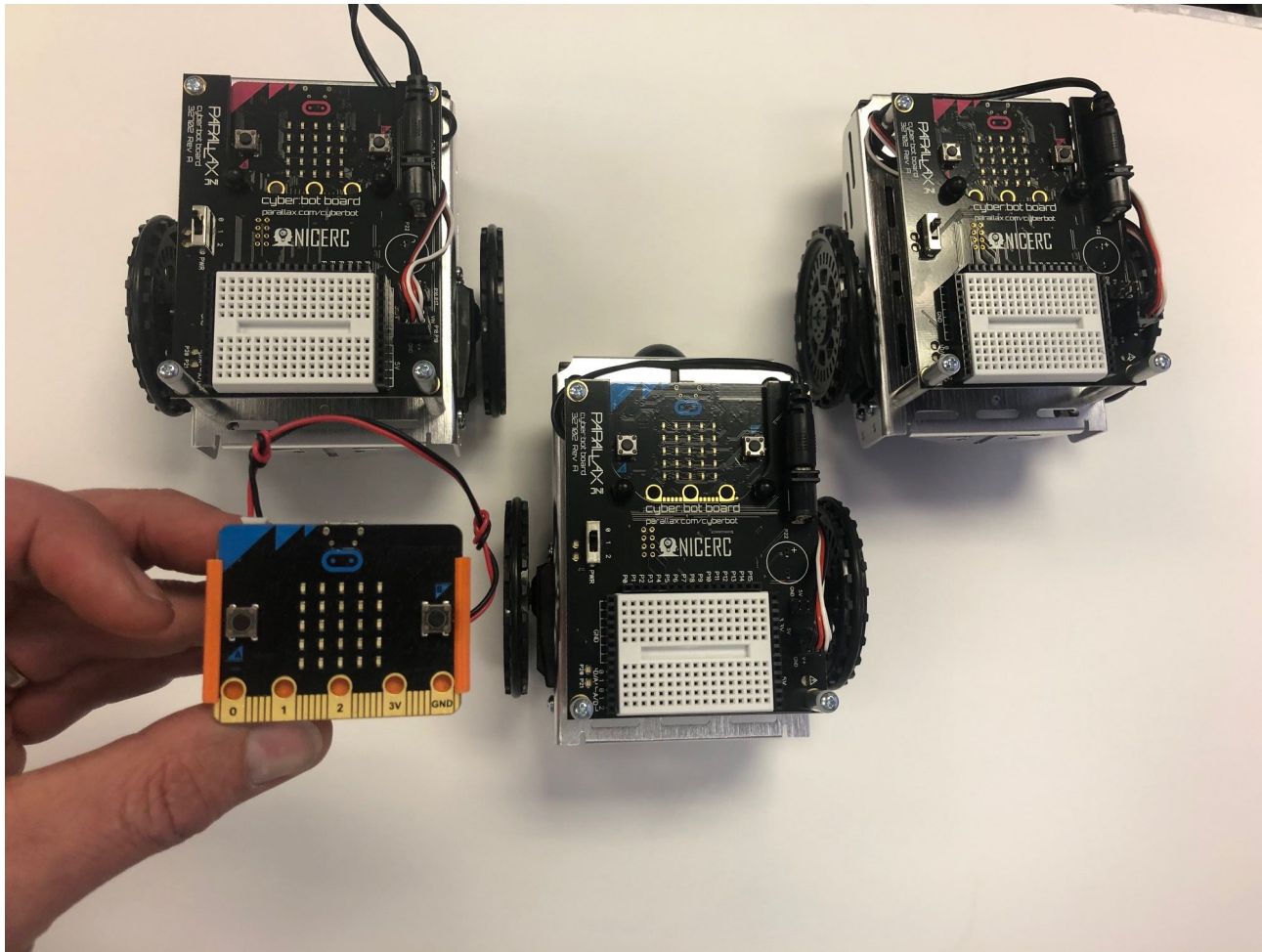


# Cybersecurity Tutorial Topics

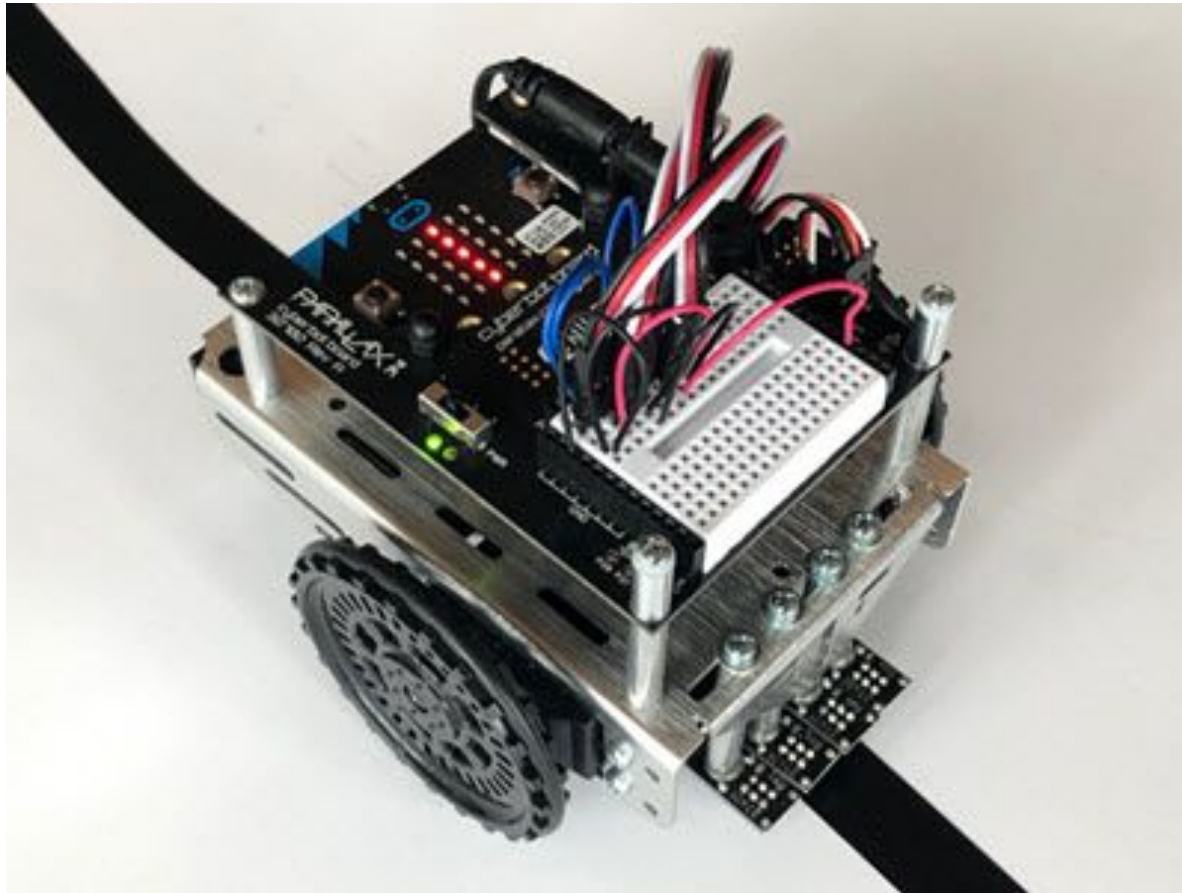
- Computer — micro:bit Talk
- Strings, Dictionaries, and Error Handling
- Radio Basics
- Radio Navigation Control
- Simple Attacks and Detections
- Denial of Service
- Authentication and Encryption
- Brute Force
- Key Exchange
- Replay Attack & Defense



# In Development: Bluetooth RF Control



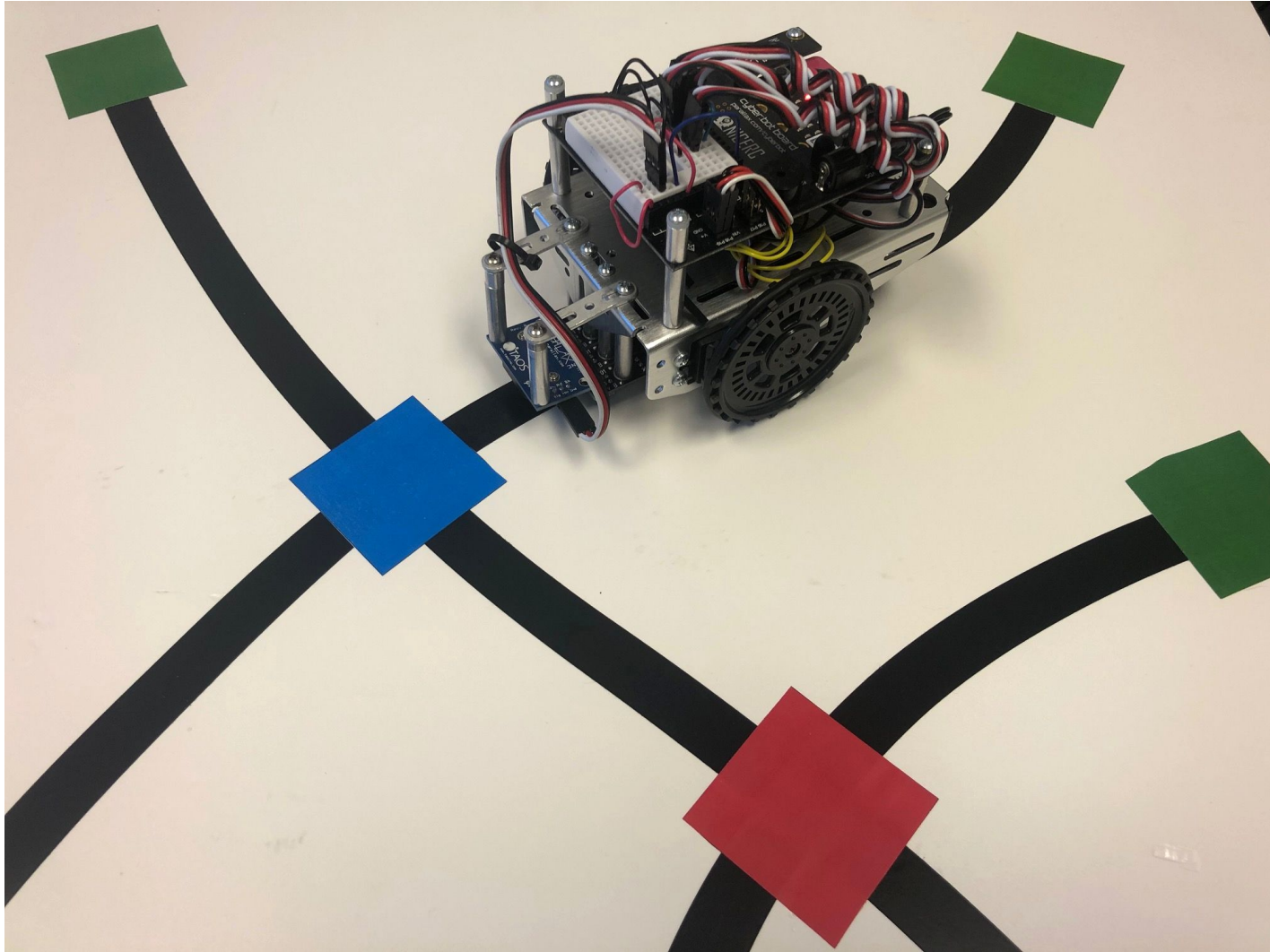
# Line Follower



<http://learn.parallax.com/tutorials/robot/cyberbot/qti-line-follower-cyberbot>



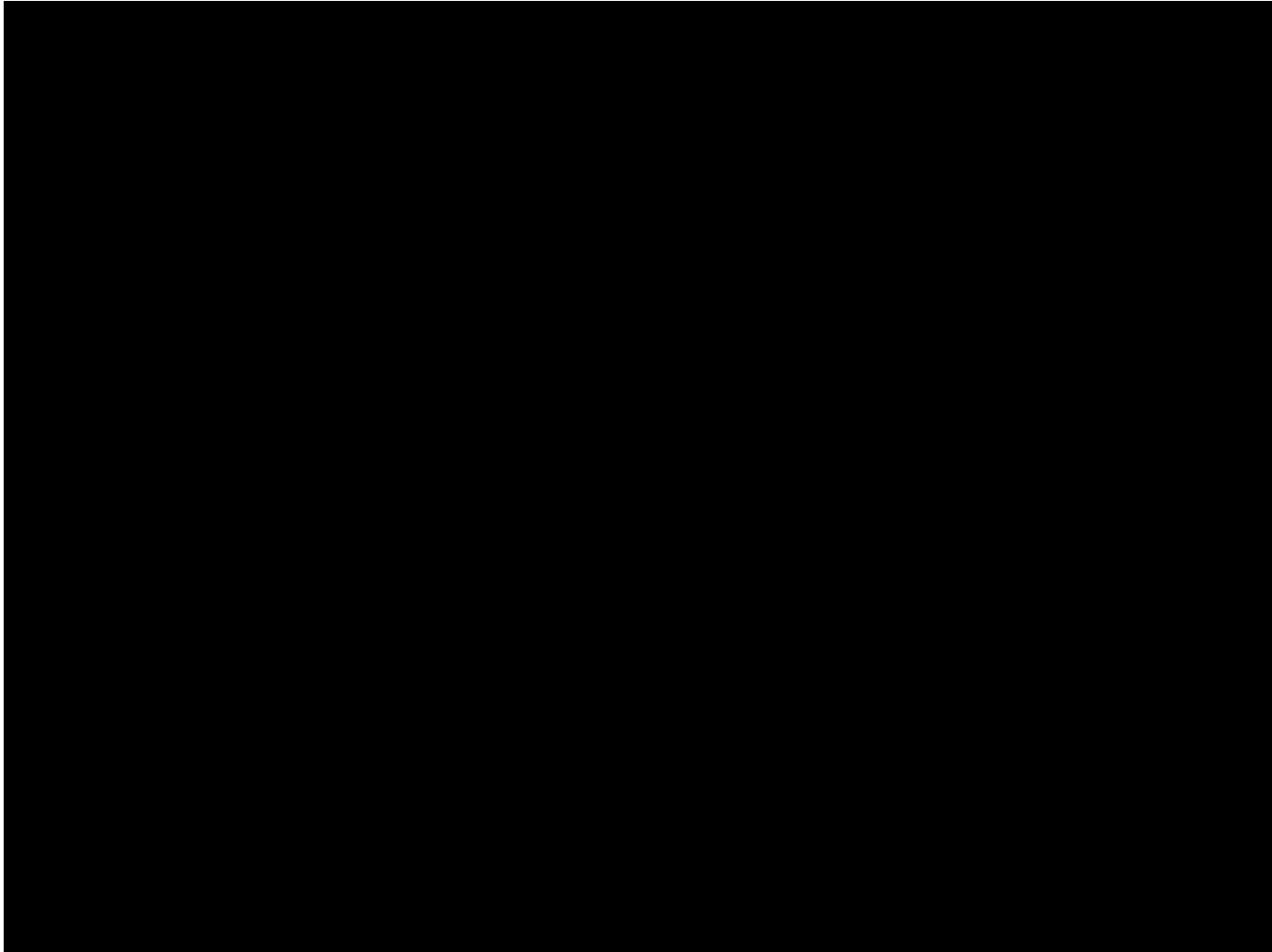
# In Development: Line Following with Color Codes

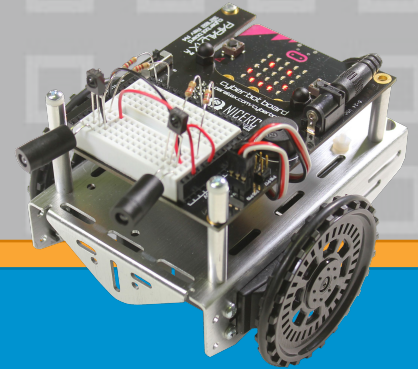


**NICERC**<sup>™</sup>  
AN ACADEMIC DIVISION OF THE  
CYBER INNOVATION CENTER



# In Development: Gripper





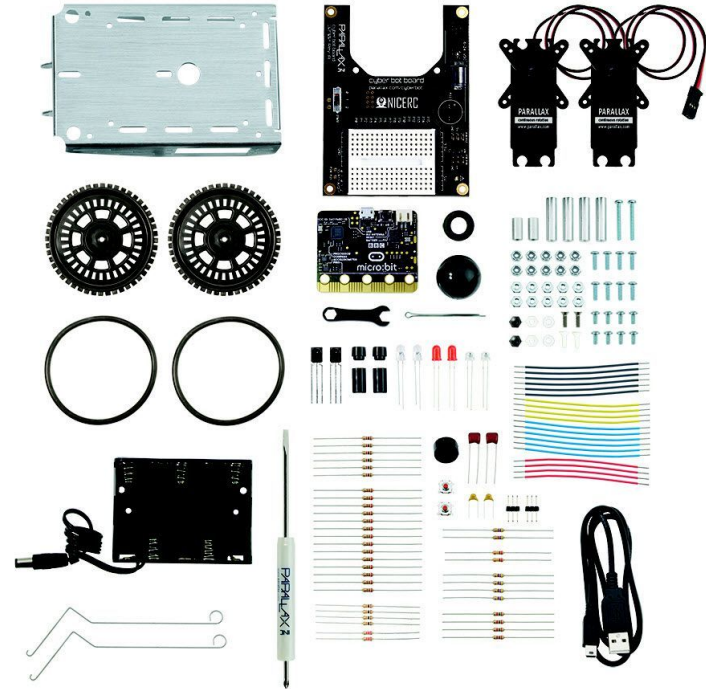
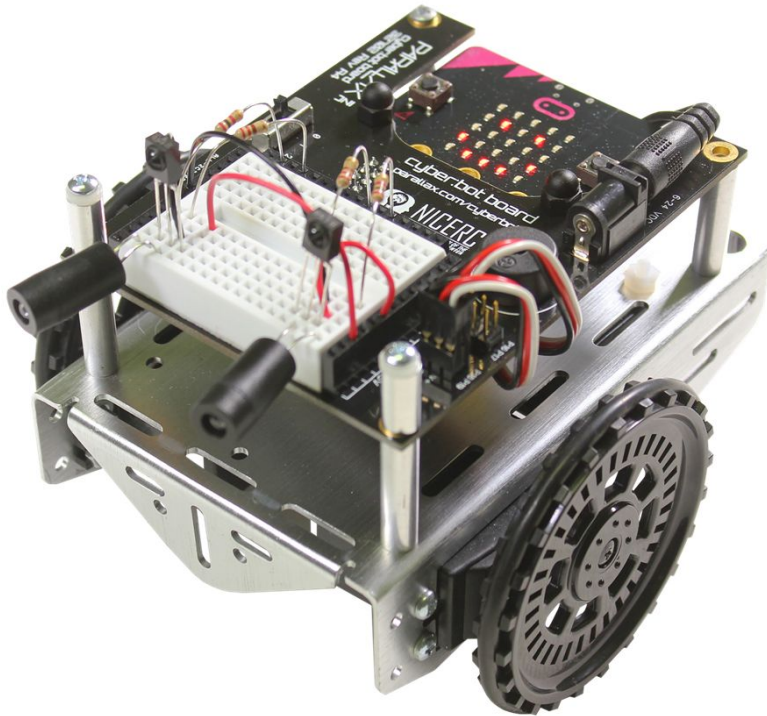
# Purchasing and Support

# Cyber:bot with micro:bit #32700

\$219 ea. (qty 1-9)

\$208.05 ea. (qty 10-19)

**\$179.10 ea. (qty 20+)**



Also available at [www.mouser.com](http://www.mouser.com) #619-32700



**MOUSER**  
ELECTRONICS®



**NICERC**  
AN ACADEMIC DIVISION OF THE  
**CYBER INNOVATION CENTER**



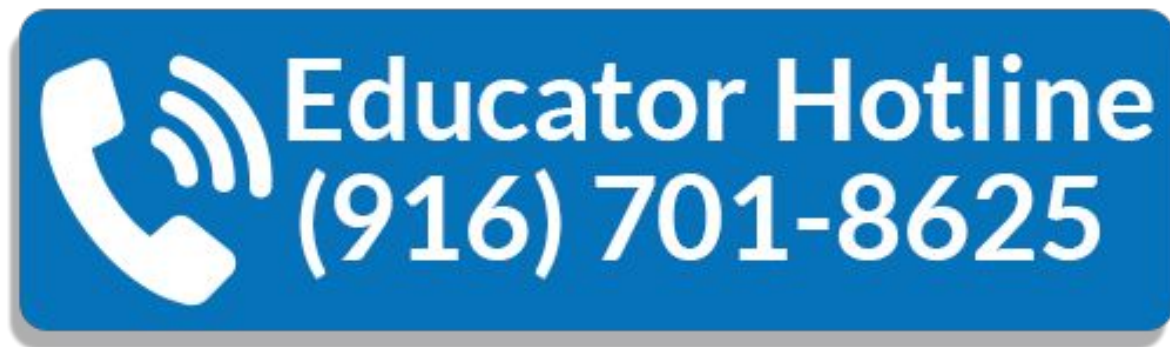


# Cyber:bot 12-Pack Plus #32712

**\$3,595** (regular \$4,330)

- (12) cyber:bot kits
- (12) micro:bit modules
- (12) QTI Line Followers
- (12) Ping))) Ultrasonic sensors and servo mounting brackets
- (12) Infrared remote controls
- (5) battery chargers
- (120) NiMH batteries
- 2'x6' class banner

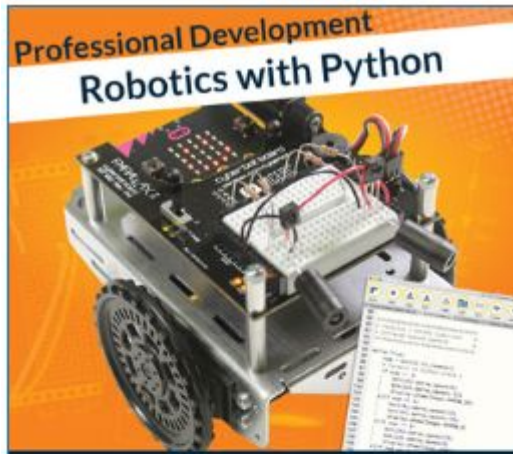




- Educator Hotline open 12 hrs/day (916) 701-8625
- E-mail [learn@parallax.com](mailto:learn@parallax.com)
- Sales (916) 624-8333
- Forums <http://forums.parallax.com/>
- Facebook
  - Parallax <https://www.facebook.com/ParallaxInc/>
  - Micro:bit <https://www.facebook.com/groups/1756471244599979/>

# Cyber:bot Workshop for Educators in Rocklin, CA: Thursday February 13

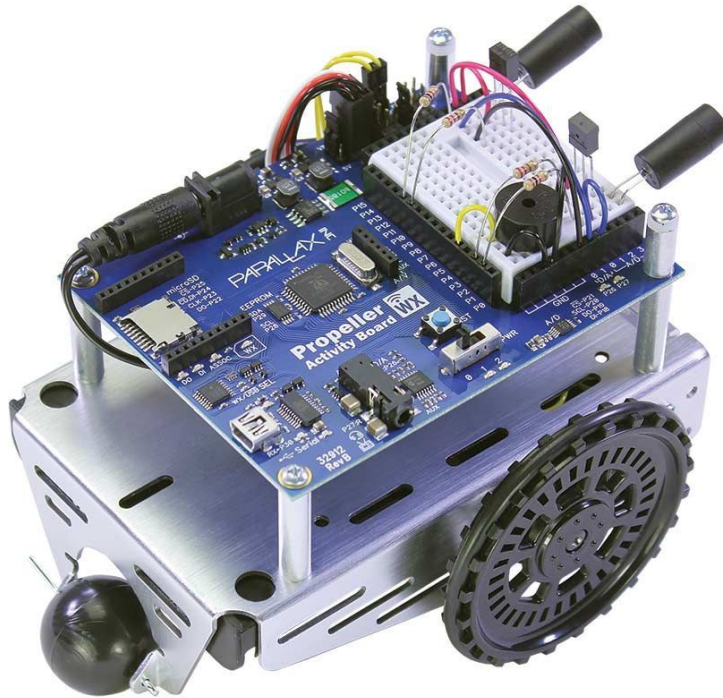
- Full day of training
- No cost (free!)
- Includes cyber:bot
- Limit 25 educators
- Register at [parallax.com/events](http://parallax.com/events)





# BlocklyProp and the ActivityBot 360 Webinar: Tuesday, January 21st 1:00 pm (Pacific)

- Register at [parallax.com/events](http://parallax.com/events)



```
add comment Light Roaming
frequency PIN 4 duration (ms) 1000 frequency (Hz) 3000
Robot ActivityBot 360 initialize
Robot set acceleration for speed blocks to 600 ticks/s² (peppy)

repeat forever
do
  add comment Read light sensors
  make PIN 9 high
  pause (ms) 1
  lightLeft = RC time PIN 9 discharge
  make PIN 5 high
  pause (ms) 1
  lightRight = RC time PIN 5 discharge

  add comment calculate nDiff
  numerator = 200 * lightRight
  denominator = lightRight + lightLeft
  nDiff = numerator ÷ denominator
  nDiff = nDiff - 100
  nDiff = nDiff * 4

  add comment calculate wheel speeds
  speedLeft = 64
  speedRight = 64
  if nDiff > 0
  do
    speedLeft = 64 - nDiff
  else
    speedRight = 64 + nDiff

  Robot drive speed
  left speedLeft
  right speedRight
```



A low-angle photograph of a blue building facade. The word "PARALLAX" is mounted on the wall in large, white, three-dimensional block letters. Below the letters is a modern architectural structure made of grey metal beams. Two spherical outdoor lights are visible on the wall. The sky is a clear, pale blue.

# PARALLAX

thank you!

**Parallax Inc.**

599 Menlo Drive, Ste 100  
Rocklin, CA 95765

[www.parallax.com](http://www.parallax.com)

[Learn.parallax.com](http://Learn.parallax.com)

Main: (916) 624-8333

Educator Hotline: (916) 701-8625

A photograph of the National Integrated Cyber Education Research Center (NICERC) building at night. The building features a curved facade and several tall, illuminated, angled concrete pillars that support a cantilevered upper section. The interior lights are visible through the glass walls, and the sky is a deep blue.

thank you!

**National Integrated Cyber Education  
Research Center, NICERC**  
6300 East Texas Street  
Bossier City, LA 71111

[www.nicerc.org](http://www.nicerc.org)  
Main: (318) 759-1600