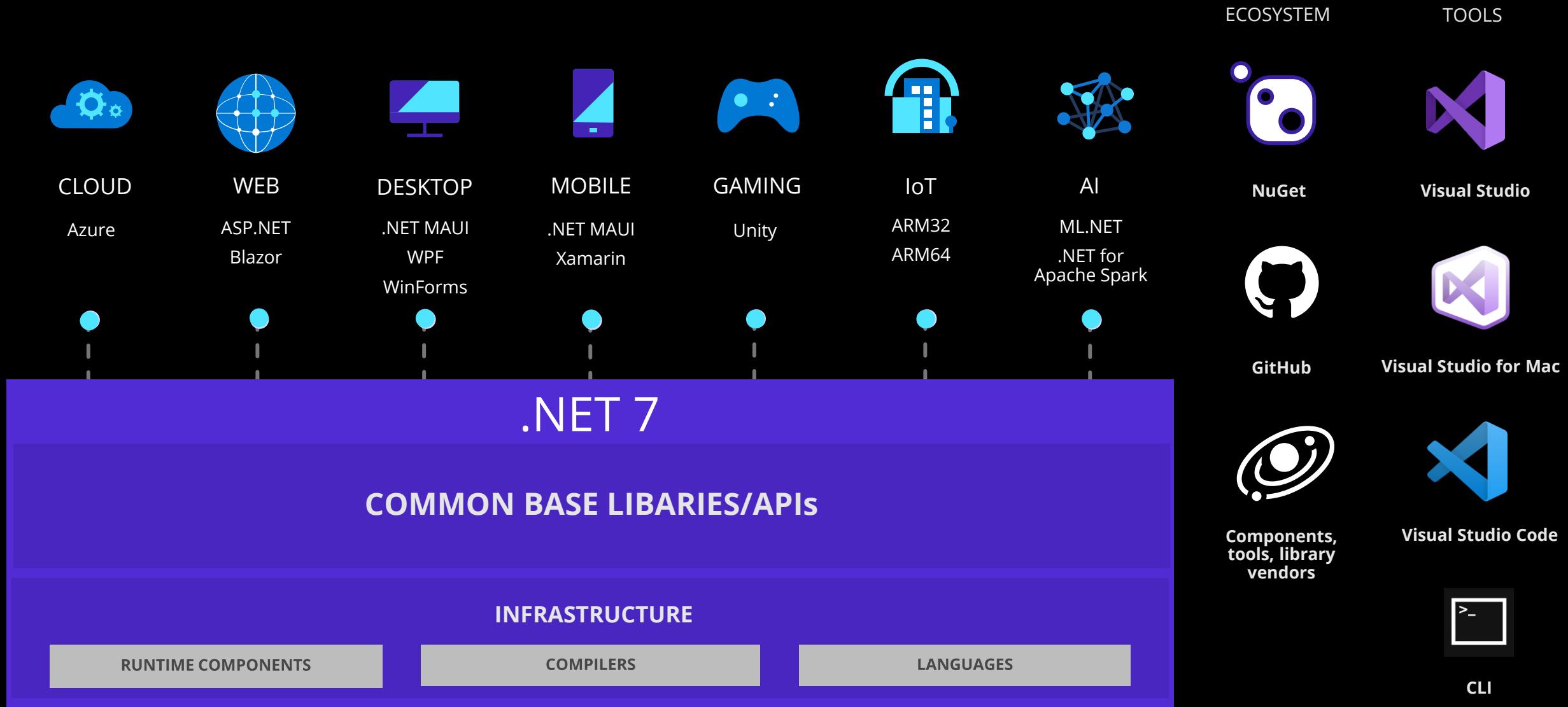




.NET IoT and AI Hack Days - Session 2



.NET - A unified development platform



What is Session 2 about?

- Getting started with Raspberry Pi and basic electronics – recap
- Quick hands-on recap on Azure IoT Central
- Learning about HaT's
- Continuing learning about electronics
- Building A Raspberry Pi robot platform
- Controlling the robot with .NET



Foundation for more complex sessions

Recap of Session 1

- What .NET is
- Devices
 - Microcontrollers
 - Microcomputer SBCs
- Hardware Interfacing
 - I2C
 - GPIO
- Module 1
 - Running a .NET Application locally and remotely on a Raspberry Pi
 - Installing .NET 7
 - Connecting to a Raspberry Pi via SSH

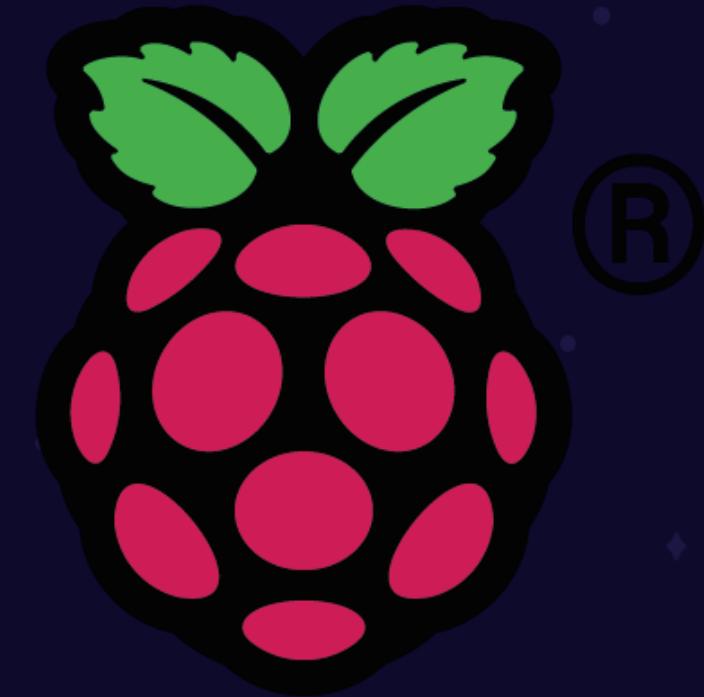
Recap of Session 1

- Module 1
 - Running a .NET Application locally and remotely on a Raspberry Pi
 - Installing .NET 7
 - Connecting to a Raspberry Pi via SSH – PUTTY
 - Remoting to a Pi with VNC
 - WINSCP to copy files to Raspberry Pi
 - Running your first .NET 7 application via SSH
 - Remote compilation / cross compilation for ARM in VS Code and remote deployment to the Pi.
 - Visual Studio Remote Debugger

Recap of Session 1

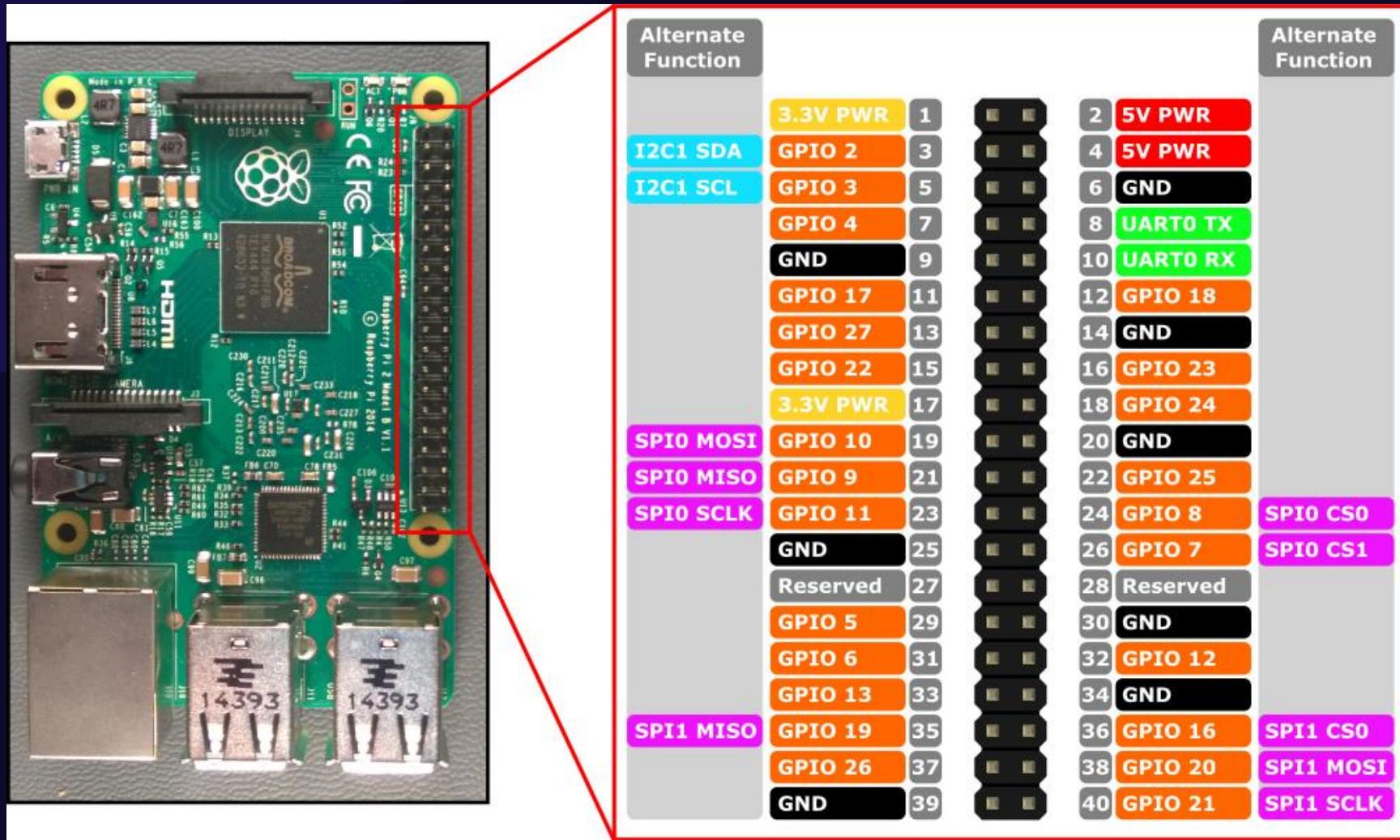
- Module 2
 - Getting started with Hardware and Sensors connected to a Raspberry Pi with .NET
 - Connecting Sensors to the Raspberry PI
 - Azure IoT and Azure IoT Central

Module 2



Getting started with Hardware and Sensors connected to a
Raspberry Pi with .NET

Interfacing to the Raspberry Pi



Interfacing to the Raspberry Pi

GPIO - General-purpose input/output

Switching things on, switching things off

Analog

No just 1 (on) and 0 (off)

Variable Input and Output voltage.

Raspberry Pi has no Analog pins. Requires an Analog to Digital Converter

I2C

Serial communication Bus

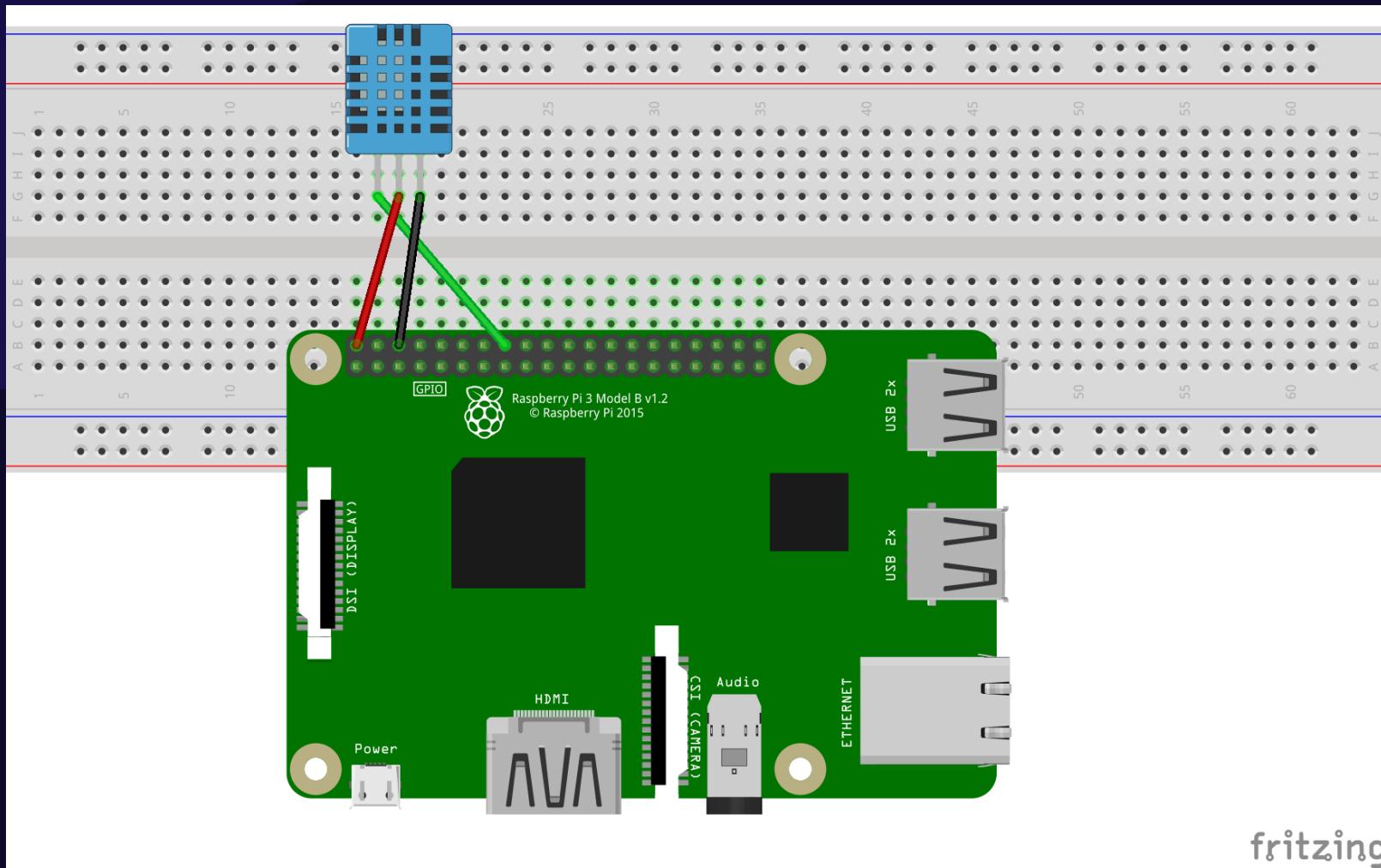
Handy as it's usually 2 wires for complex data communication

SPI (Serial Peripheral Interface)

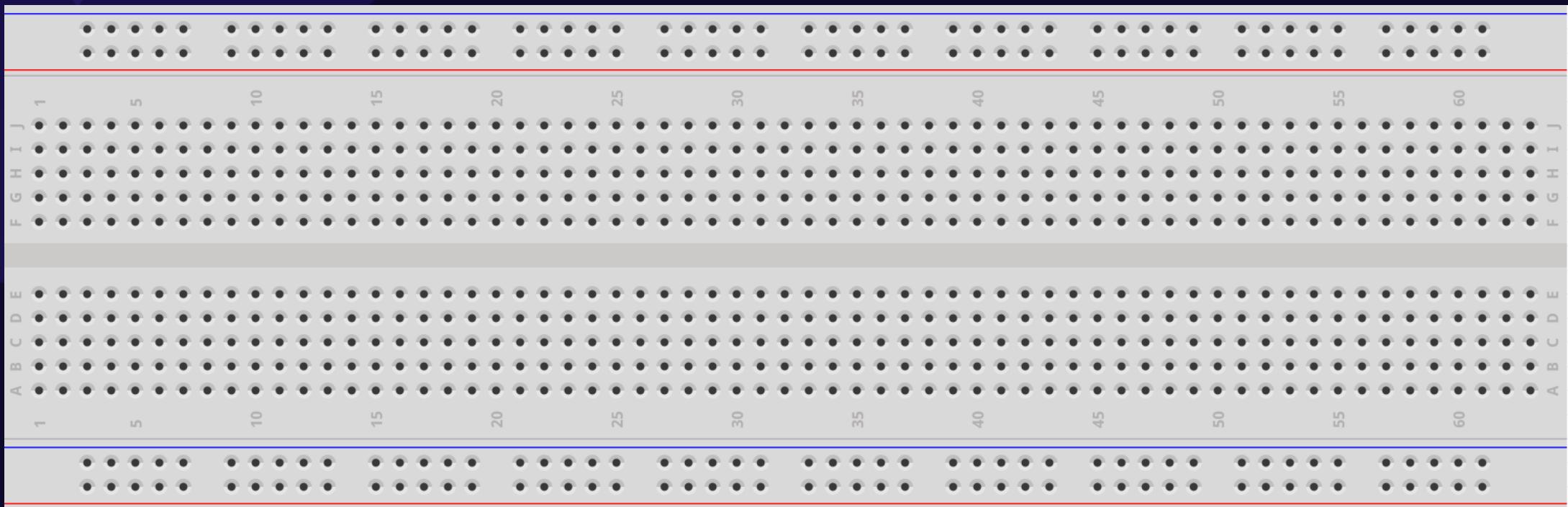
Serial communication bus

Usually 3 wire communication

Let's Build an Environment Monitor

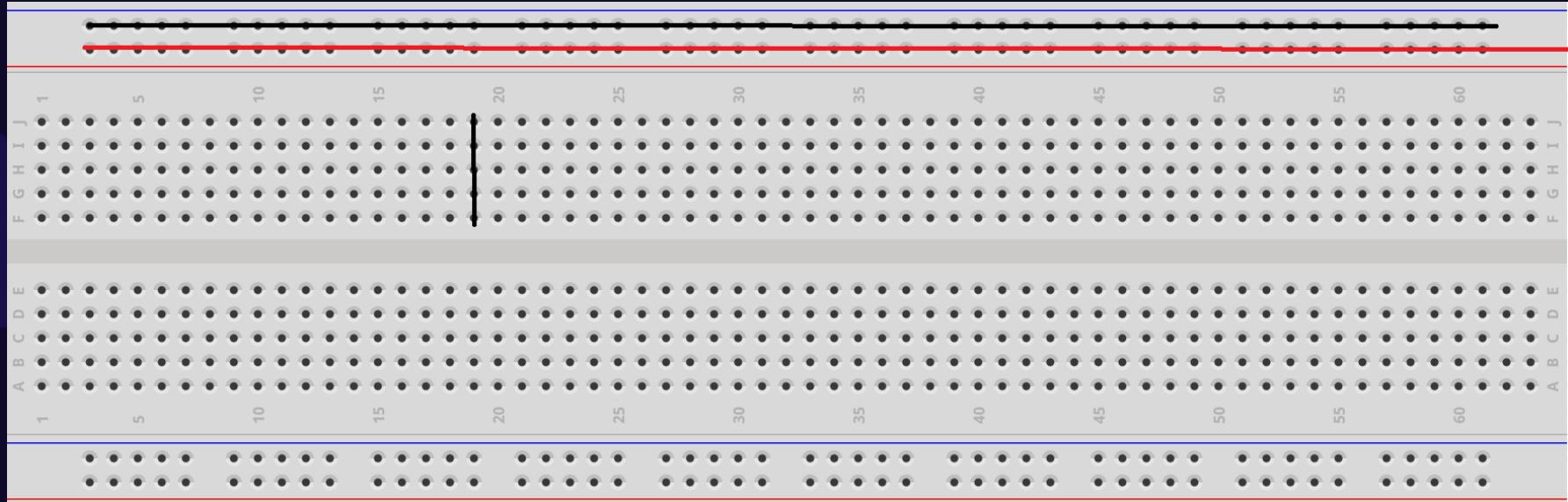


The breadboard



The breadboard

Pins Connected Horizontally



Pins connected vertically

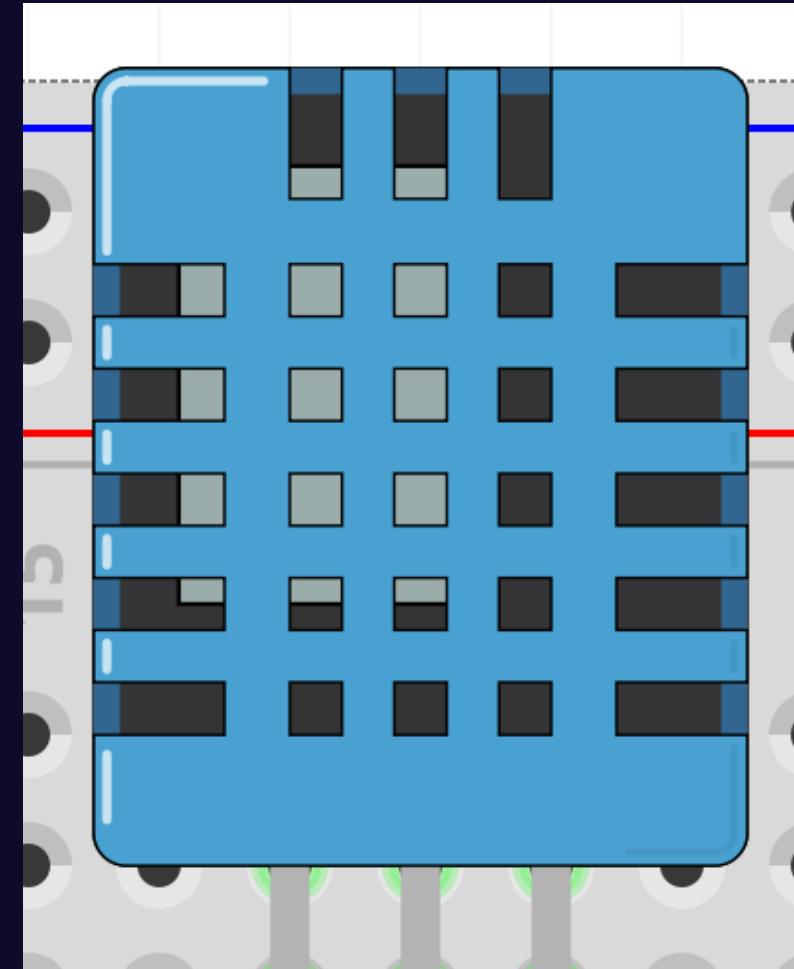
Pins Connected Horizontally

Used only for experimenting and prototyping

DHT 11

Temperature and humidity
sensor

Serial Data



DHT 11

- Pin 2 - Vcc

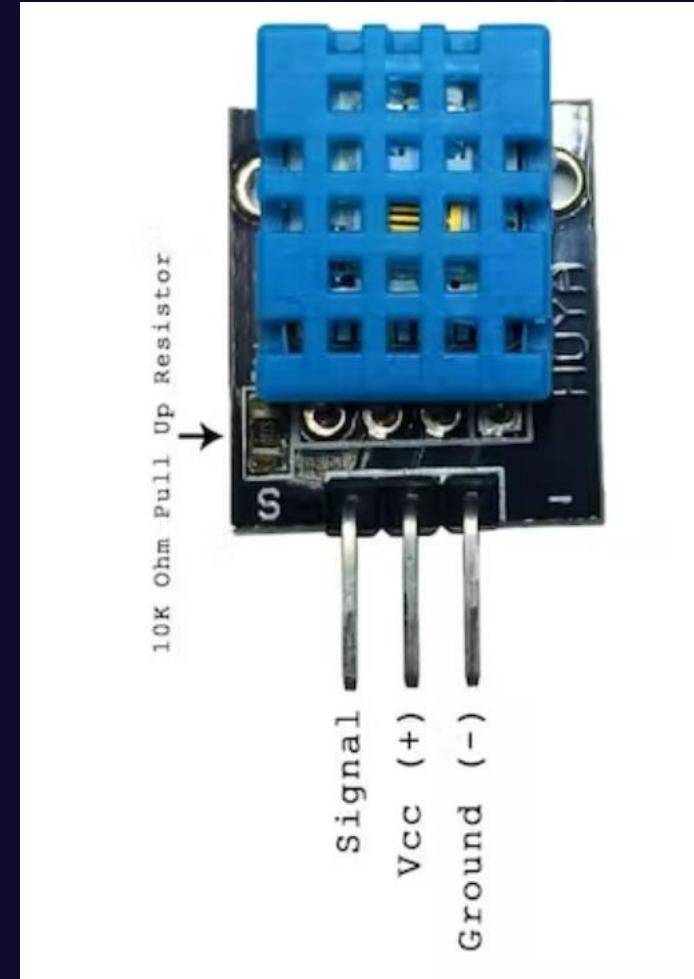
Power supply 3.5V to 5.5V

Pin 1 - Data

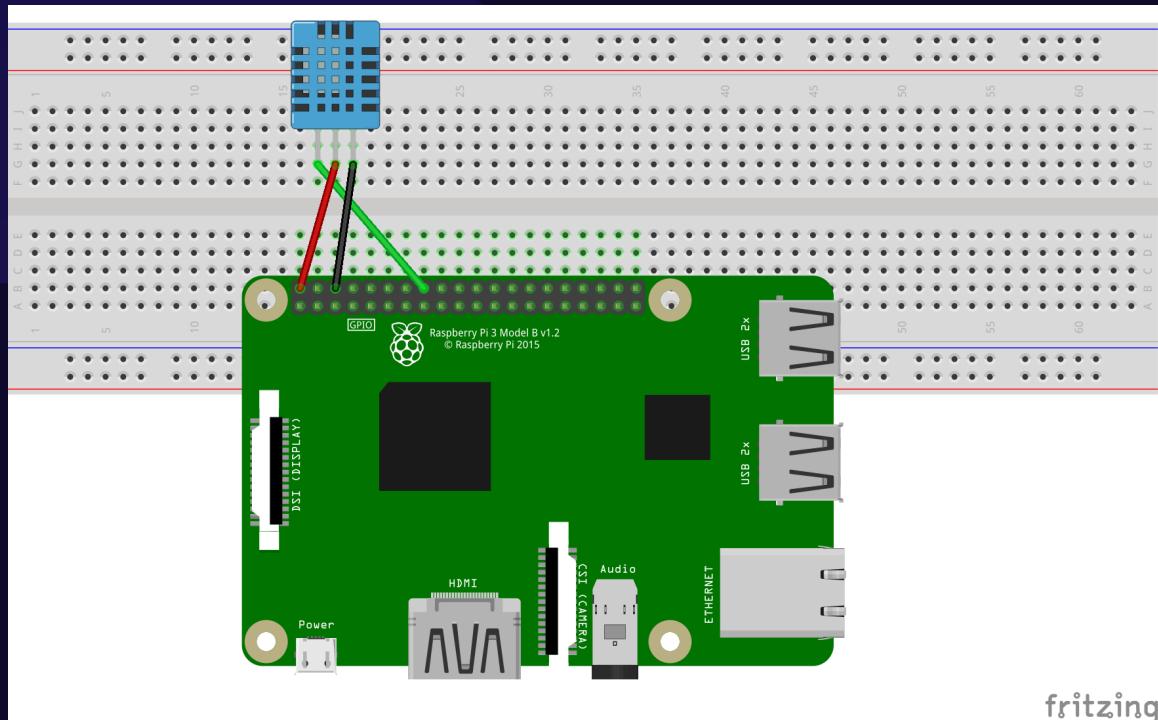
Outputs both Temperature and Humidity through serial Data

Pin 3 - Ground

Connected to the ground of the circuit



Try it out!



Using the component pack, build the circuit using your Raspberry Pi, breadboard, DHT11 and wires.

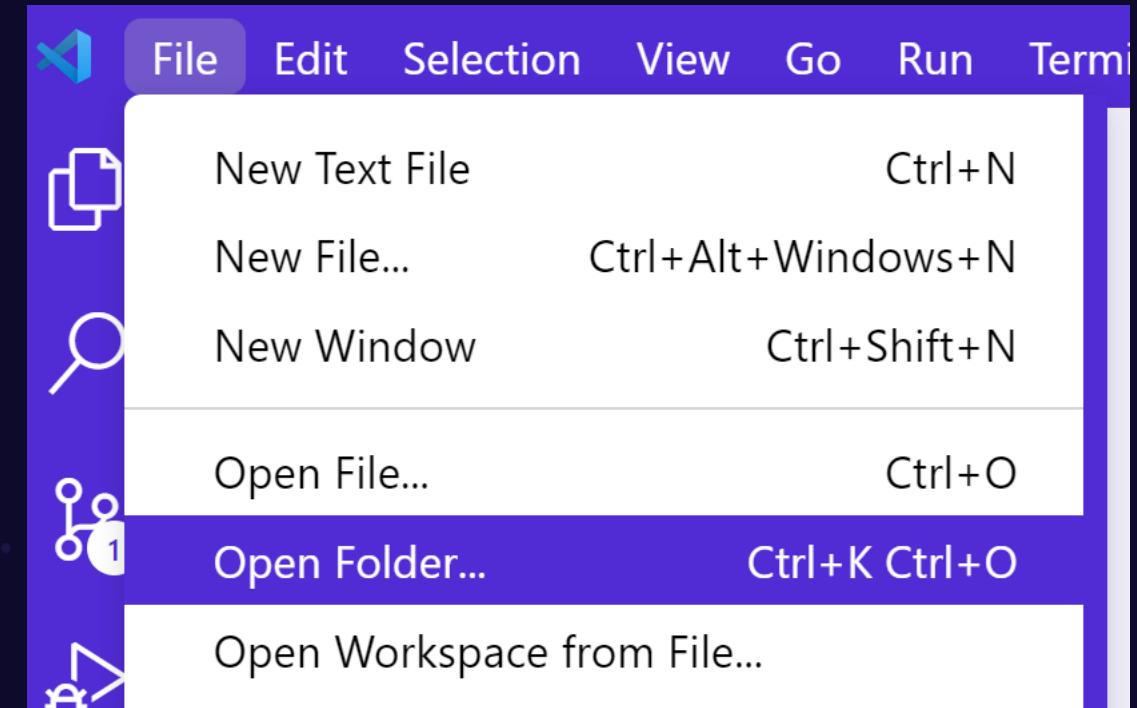
5V from the Raspberry Pi to the middle pin of the DHT 11 Sensor (VCC)

Ground from the Raspberry Pi to the right pin of the DHT Sensor (Ground)

GPIO Pin 23 from the Raspberry Pi to the left pin on the DHT Sensor (Data / Signal)

Try it out!

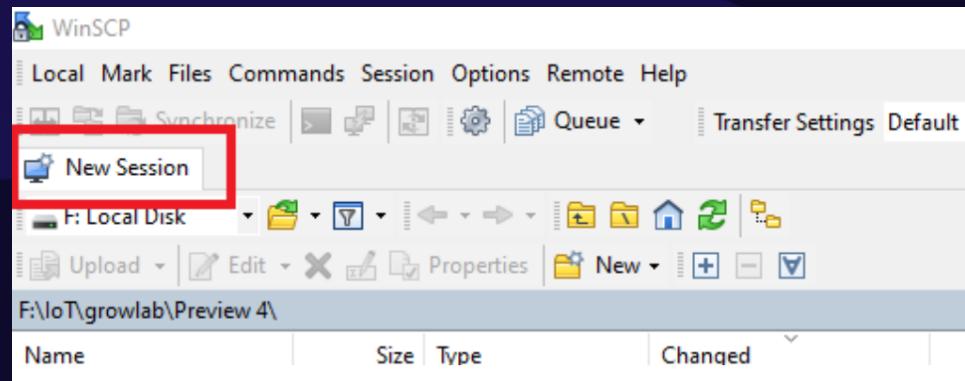
**In VS Code open the folder
SensorEnd from the code cloned
from Github**



Git clone <https://github.com/CPTMSDUG/IoTHackDaysSession2DotNet>

Ensure WinSCP is set up!

WinSCP



Try it out!!

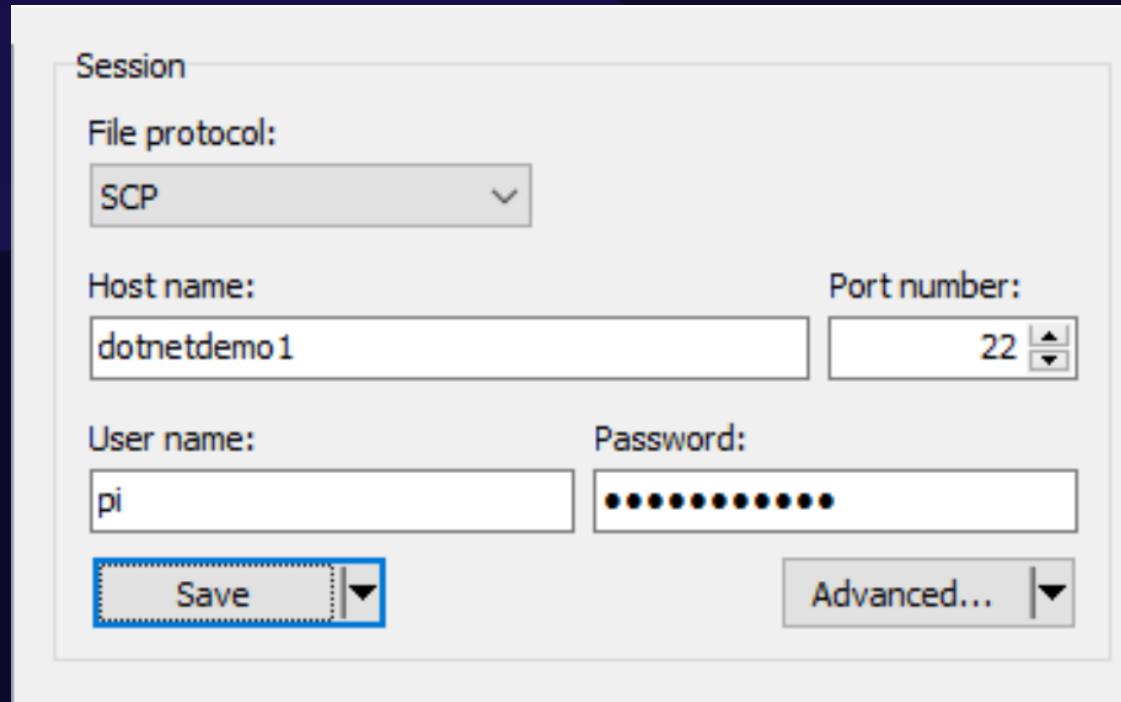
Switch your Raspberry PI on

Run WinSCP

Create New Session

Development Tools

WinSCP



Protocol SCP

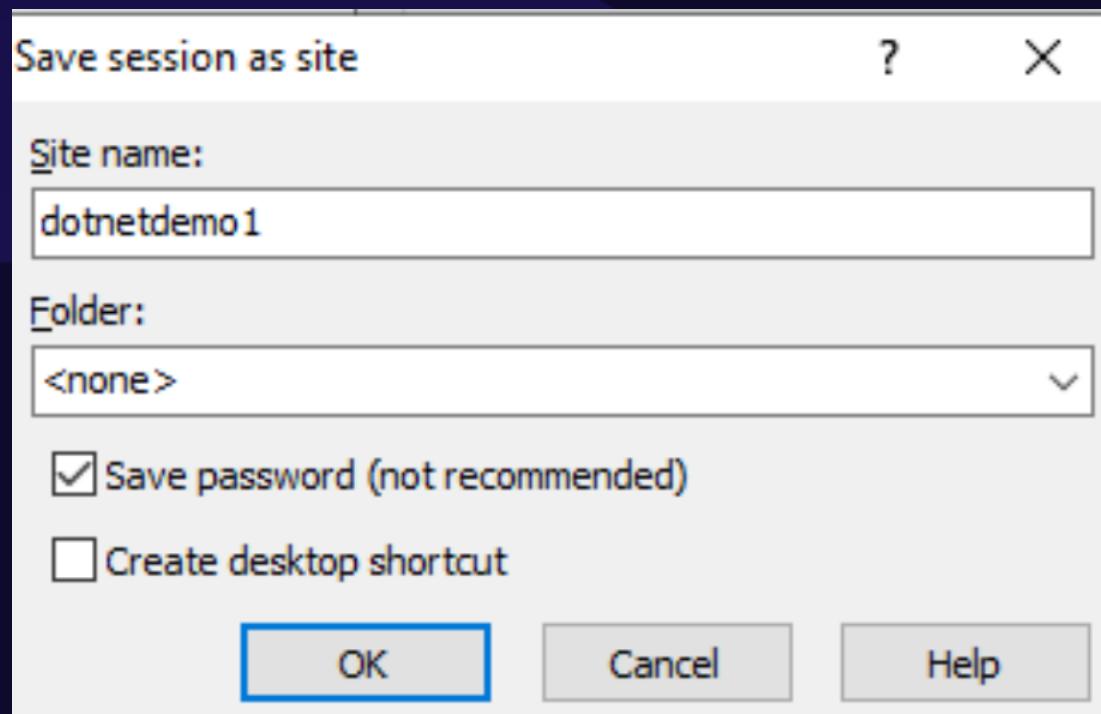
Host name: your ipaddress
(on wired network use hostname – sticker on Pi)

Username: pi

Password: mxchip12345

Development Tools

WinSCP



Save

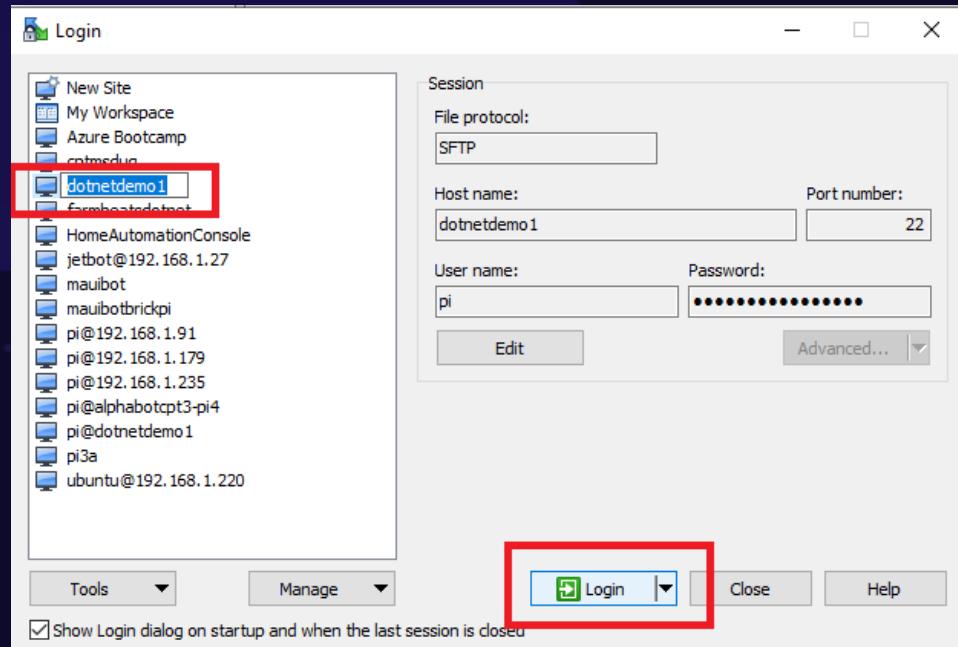
Site name: The name of your pi. Use the sticker on the pi, eg: E1163323

Save Password

Remember this name for the .env file (WinScpSite)

Development Tools

WinSCP



Login

Choose the name of your pi from the sites.

Click login

Try it out!

Don't forget to edit the .env file
with the details of your Raspberry
Pi

On wired network, use your host name
(the sticker on the Raspberry Pi)



```
.env
1 PiHostName=pi@yourip
2 PiPassword=mxchip12345
3 WinscpSite=dotnetdemo1
4
```

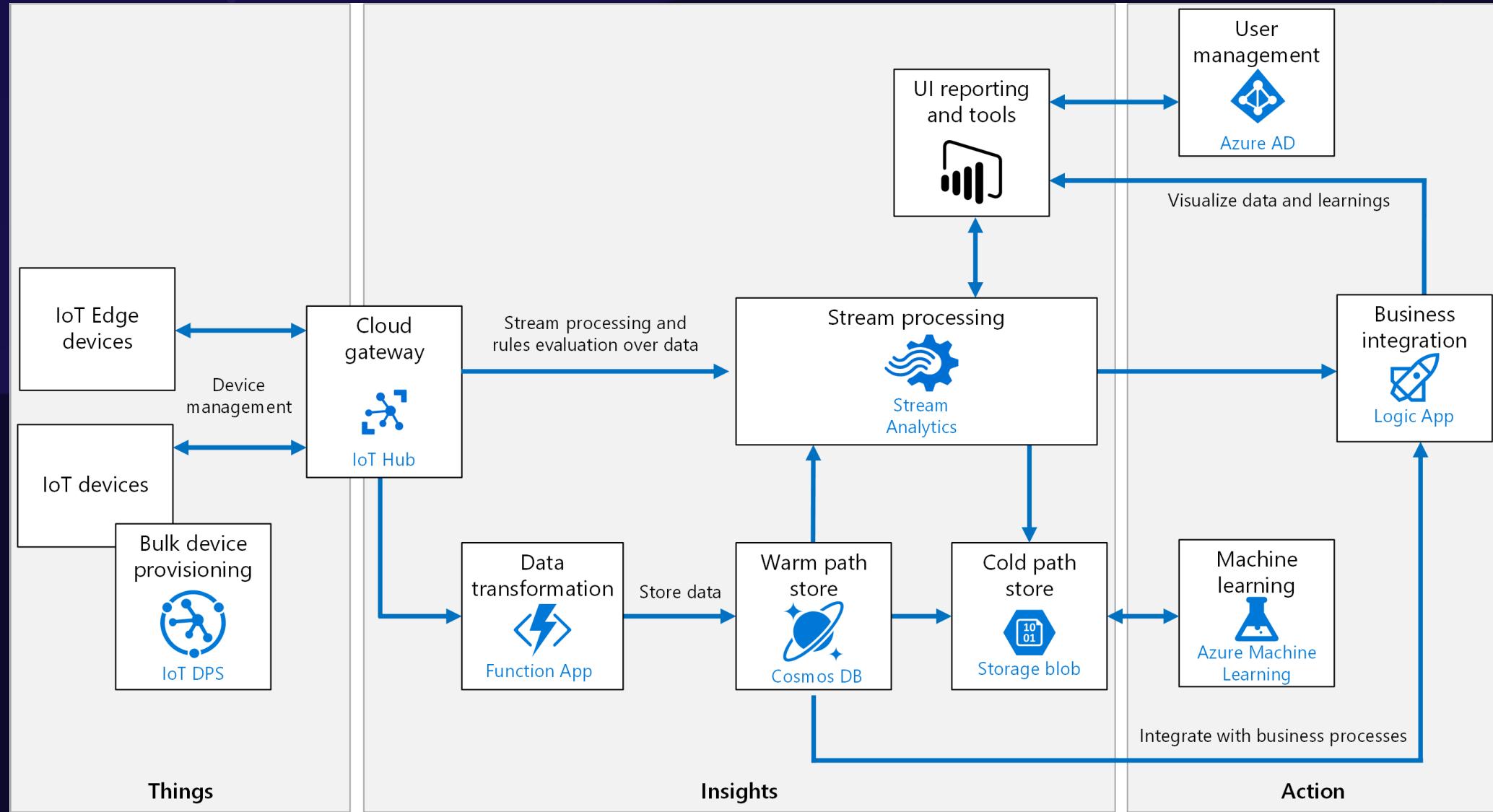
Module 3



Azure IoT Central

Getting Started with Azure IoT with Azure IoT Central

Azure IoT Reference Architecture



IoT Solutions - IoT Central



Azure IoT Central

Experience the simplicity of SaaS for IoT, with no cloud expertise required



Fast and easy

- Start in minutes and create a finished solution in hours
- Build without any cloud development expertise



Scalable and secure

- Connect your devices at any scale without worrying about infrastructure
- Get best-in-class security



Enterprise grade

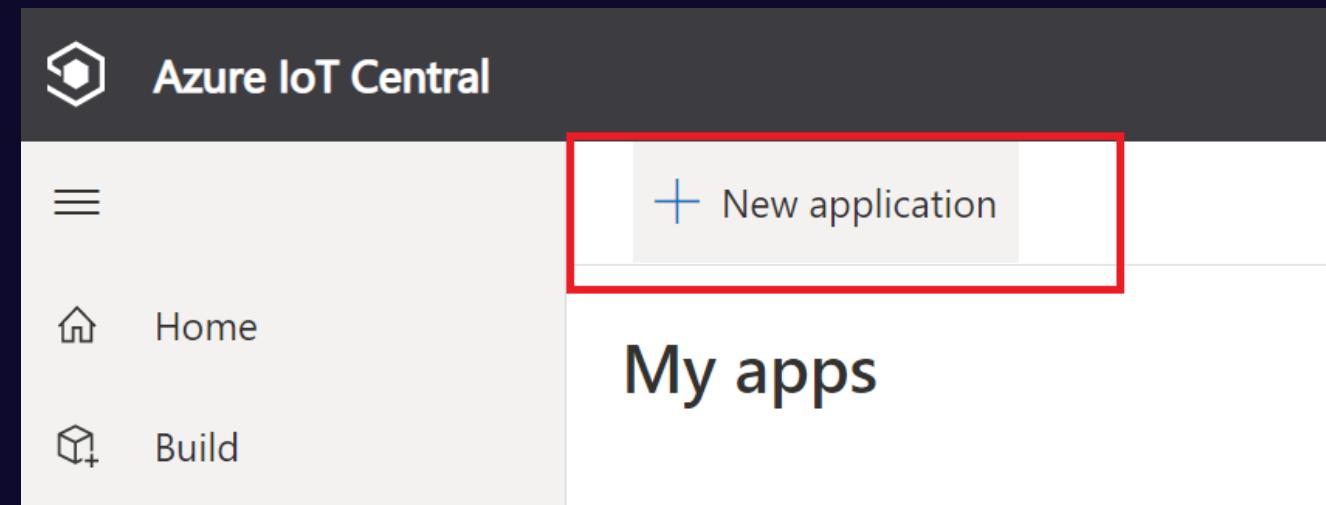
- Built on proven enterprise-grade Azure services
- Full integration into your existing business systems and processes

Try it Out!

Log into Azure IoT Central

<https://apps.azureiotcentral.com/>

Create a New Application



This Requires an Azure Subscription!

Try it Out!

Create Custom App

Build your IoT application

Featured



Custom app

Create a custom application to build a unique solution for your business using powerful tools to connect, monitor, and manage your IoT data.

[Create app](#)

[Learn more](#)

This Requires an Azure Subscription!

Try it Out!

Give your Application a Name

Chose the Standard 0 Price Plan (2 devices free)

The screenshot shows the 'Build > New application' screen. On the left, a sidebar has 'Home', 'Build' (which is selected), and 'My apps'. The main area is titled 'New application' with a 'Custom' tab. It says 'Answer a few quick questions and we'll get your app up and running.' Under 'About your app', the 'Application name *' field contains 'IoT Hack Session 1'. The 'URL *' field contains 'iot-hack-session-1' followed by '.azureiotcentral.com'. The 'Application template *' dropdown is set to 'Custom application'. Under 'Pricing plan', 'Standard 0' is selected, described as 'For devices sending a few messages per day' with '2 free devices 400 messages/mo'.

This Requires an Azure Subscription!

Try it Out!

Choose an Active Azure Subscription

Create the Application

Billing info

Directory * ⓘ

Azure subscription * ⓘ

Don't have a subscription? [Create subscription ↗](#)

Microsoft Azure Sponsorship

Location * ⓘ

West Europe

By clicking "Create" you agree to the [Subscription Agreement ↗](#) and [Privacy Statement ↗](#). "Standard" plans require an Azure subscription, and you acknowledge that this service is licensed to you under the terms applicable to your [Azure Subscription ↗](#).

Create Cancel

This Requires an Azure Subscription!

Try it Out!

Create a Device Template

The screenshot shows the Azure IoT Hub Device Templates blade. At the top left, the title "IoT Hack Session 1" is displayed. On the right, there is a search bar with the placeholder "Search for d...". Below the title, there is a message: "IoT Hub and DPS are updating their TLS certificates with a new M..." followed by a "New" button, which is highlighted with a red box. The main content area is titled "Device templates". On the left, a sidebar menu includes "Connect", "Devices", "Device groups", "Device templates" (which is selected and highlighted with a blue dashed box), "Edge manifests", "Analyze", and "Data explorer".

This Requires an Azure Subscription!

Try it Out!

Select IoT Device

Device templates > Create new device template

Select type

A device template is like a blueprint. It defines the characteristics and behaviors of devices that connect to your app.

Create a custom device template



IoT device
Import a capability model or build capabilities from scratch.

Azure IoT Edge
Create a template that features Azure IoT Edge and gateway scenarios.

This Requires an Azure Subscription!

Try it Out!

Provide the Template a name

Device templates > Create new device template

Customize

Device template name*

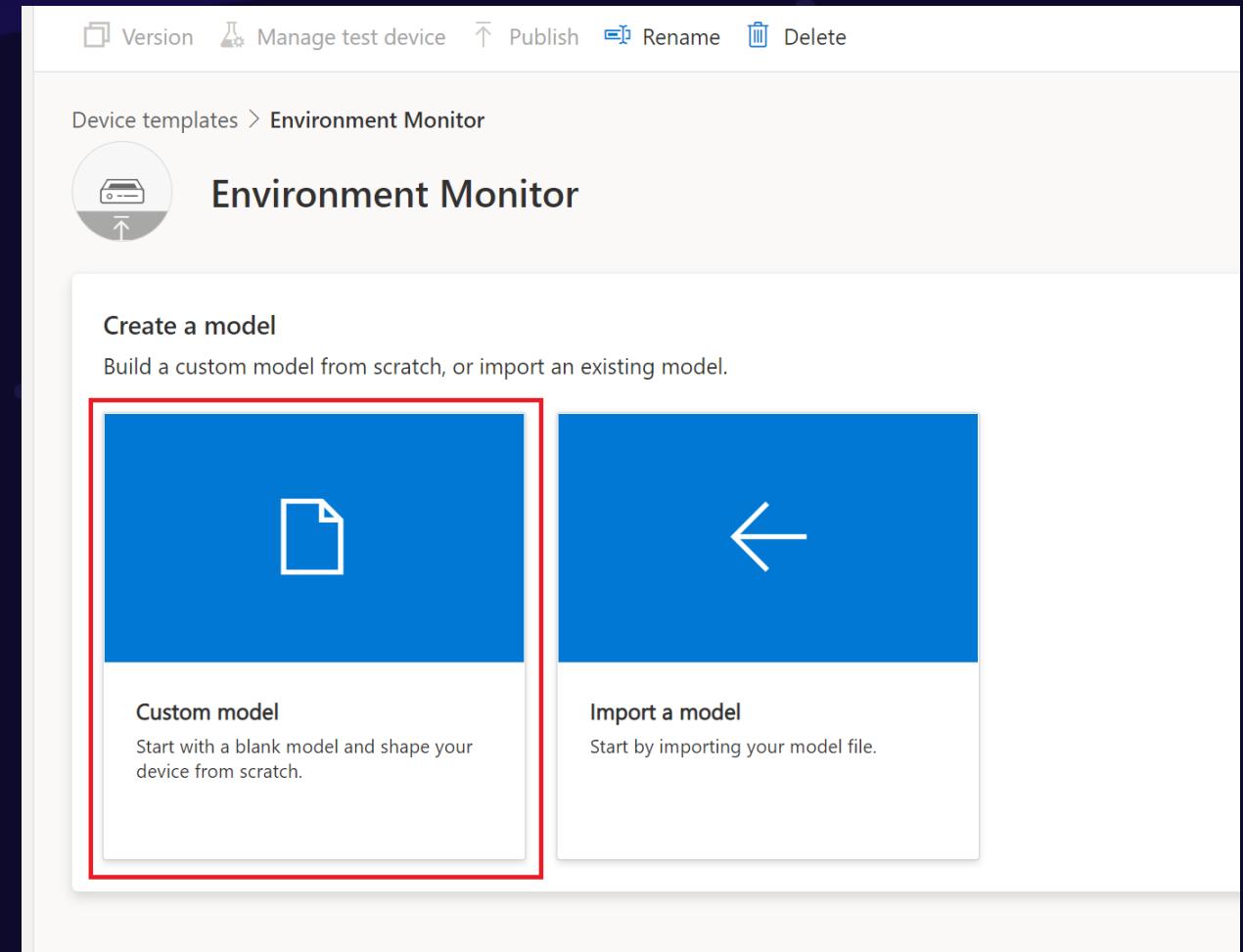
 x

This is a gateway device. [Learn more.](#)

This Requires an Azure Subscription!

Try it Out!

Create a Custom Model for
the Device Template



This Requires an Azure Subscription!

Try it Out!

Add a Capability

Device templates > Environment Monitor > Model > Environment Monitor

Environment Monitor

Application updated: Never Interfaces published: Never

> Environment Monitor **Root** Draft

Add capabilities specific to this device model. [Learn more](#)

Save + Add capability Edit identity Export Delete ...

+ Add capability

This Requires an Azure Subscription!

Try it Out!

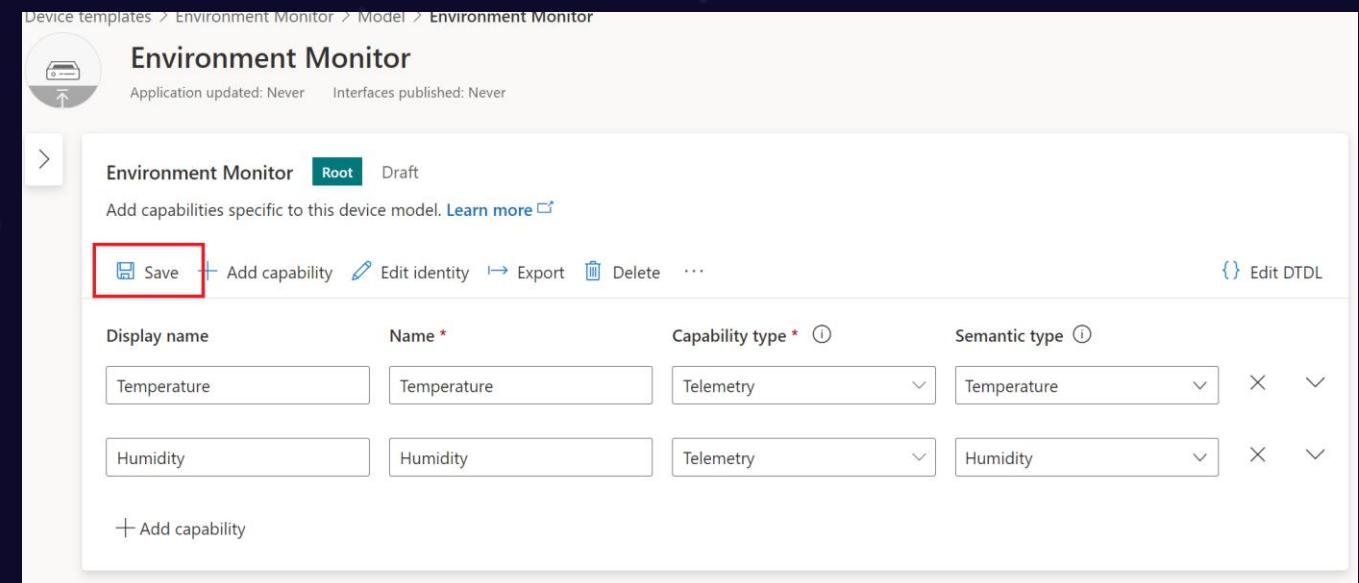
Add a Capability for Temperature

- Semantic Type Temperature

Add a Capability for Humidity

- Semantic Type Humidity

Make sure you Save



The screenshot shows the Azure Device Template Editor interface. The path in the top left corner is "Device templates > Environment Monitor > Model > Environment Monitor". The main title is "Environment Monitor" with a subtitle "Application updated: Never" and "Interfaces published: Never". Below the title, there's a button bar with "Save" (highlighted with a red box), "Add capability", "Edit identity", "Export", "Delete", and "...". To the right of the bar is a link "Edit DTDL". The main content area displays two capability definitions:

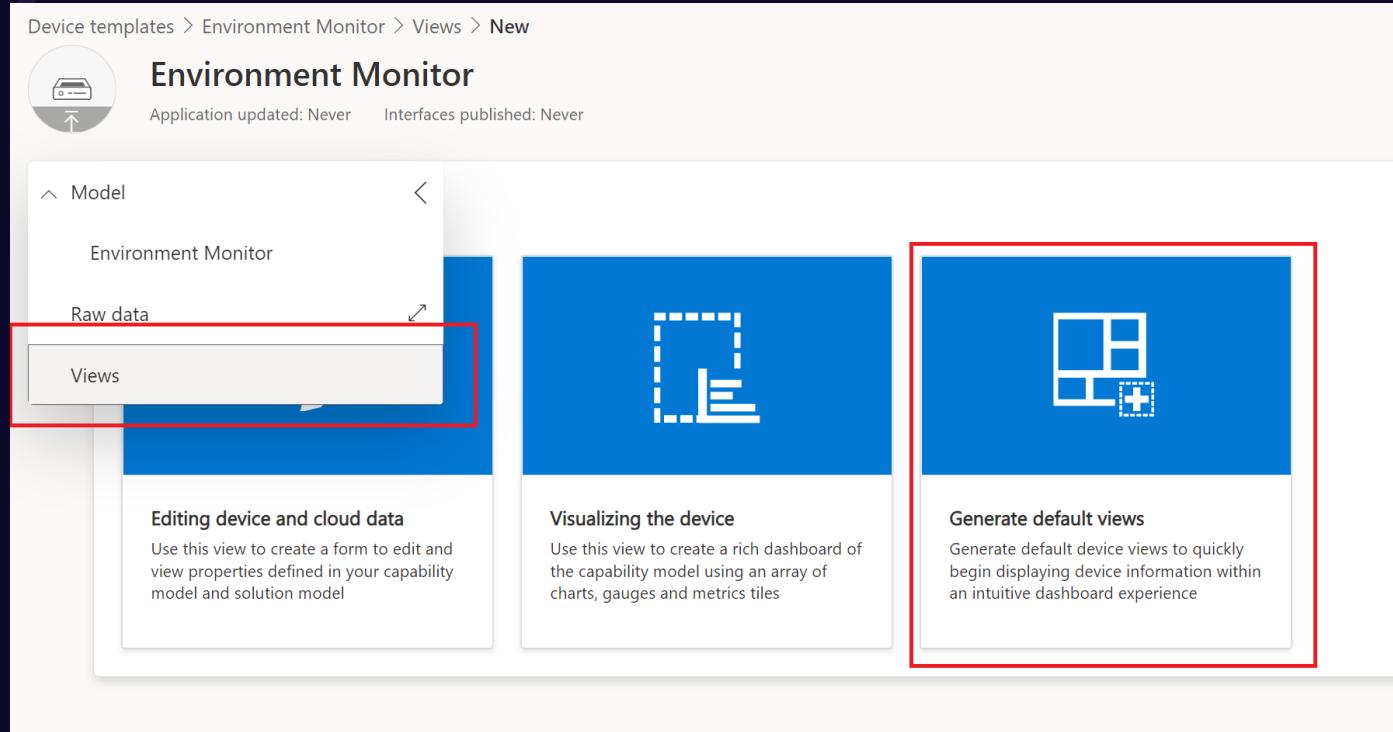
Display name	Name *	Capability type *	Semantic type
Temperature	Temperature	Telemetry	Temperature
Humidity	Humidity	Telemetry	Humidity

At the bottom of the list is a button "+ Add capability".

This Requires an Azure Subscription!

Try it Out!

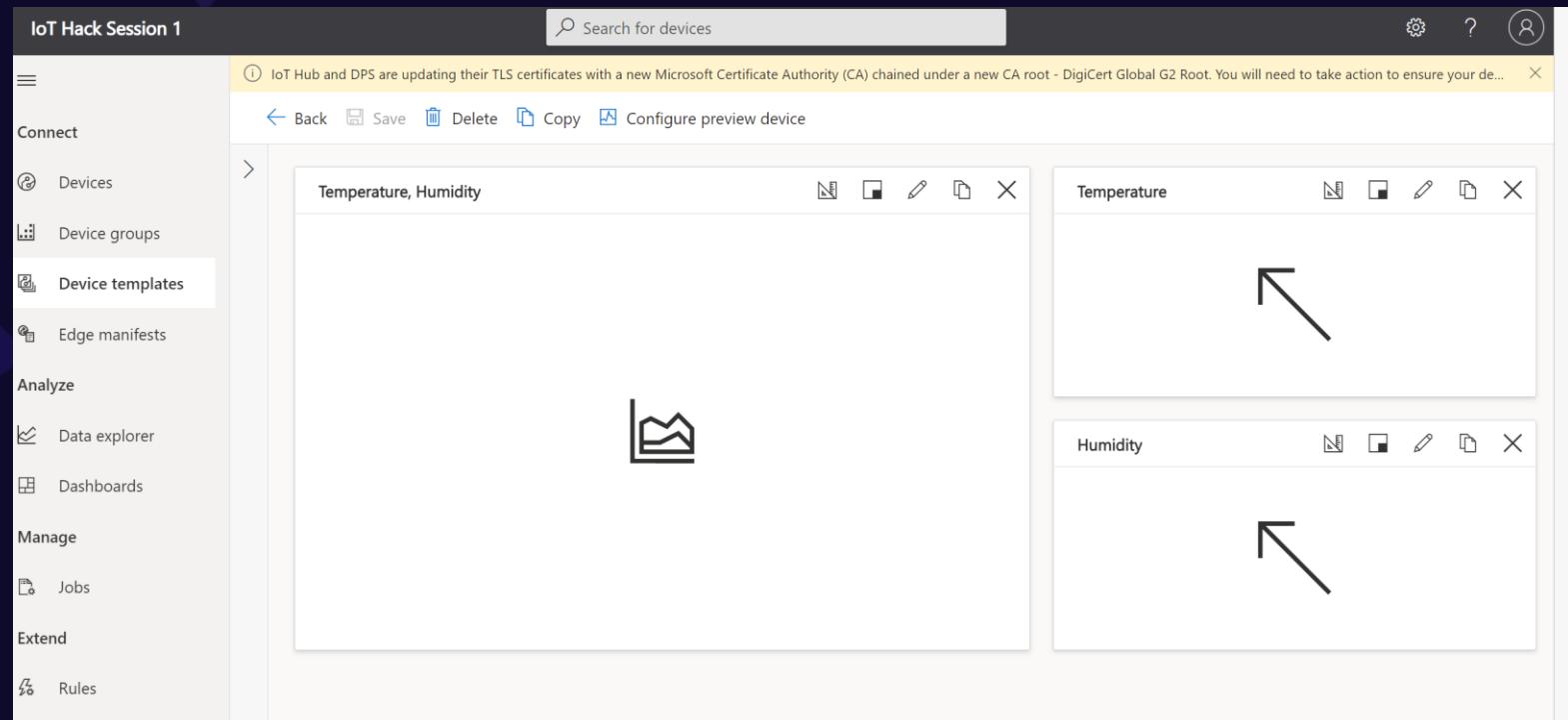
Auto-generate a view / dashboard from the model created.



This Requires an Azure Subscription!

Try it Out!

Generated Views

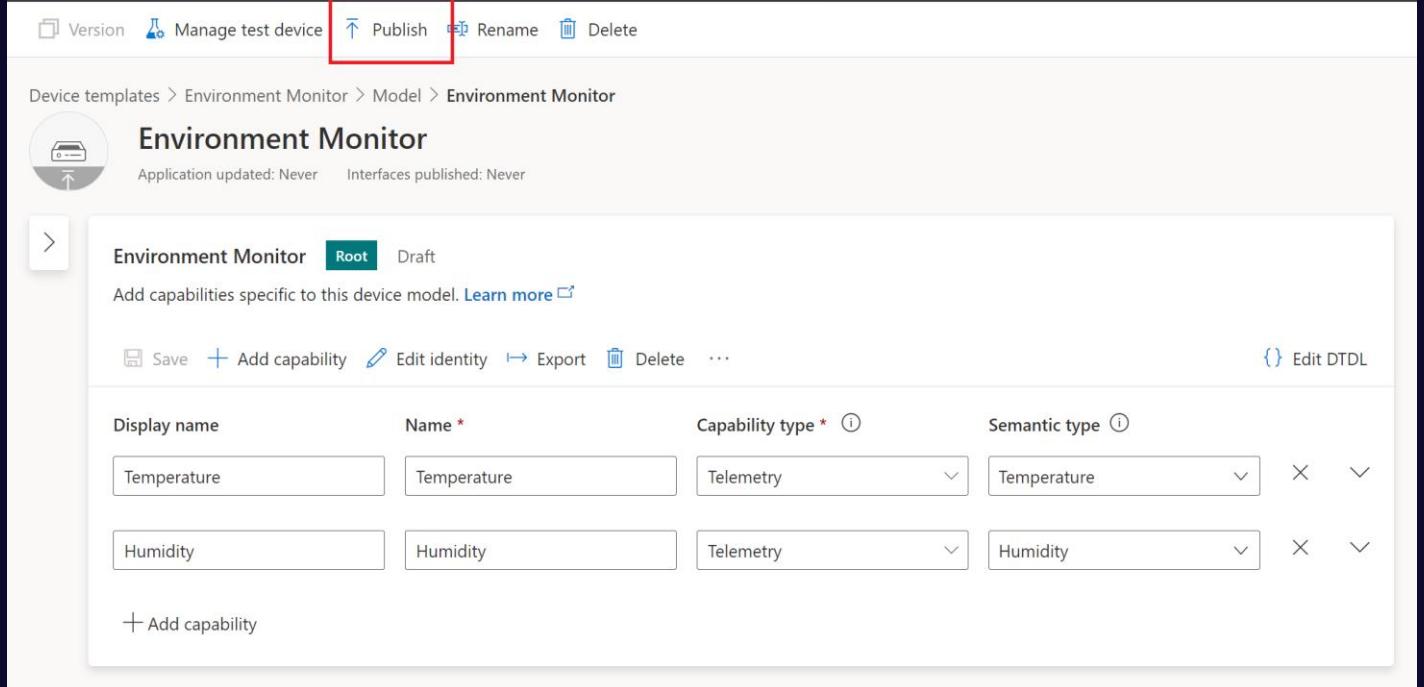


This Requires an Azure Subscription!

Try it Out!

Publish a version of the model

The versioned model can now
be used as a device.



The screenshot shows the Azure Device Template Editor interface. At the top, there's a navigation bar with 'Version', 'Manage test device', 'Publish' (which is highlighted with a red box), 'Rename', and 'Delete'. Below the navigation, the path is 'Device templates > Environment Monitor > Model > Environment Monitor'. The main title is 'Environment Monitor' with a subtitle 'Application updated: Never' and 'Interfaces published: Never'. There's a 'Save' button and a 'Root' tab. A section titled 'Add capabilities specific to this device model.' includes a 'Learn more' link. Below this, there's a table for defining device capabilities:

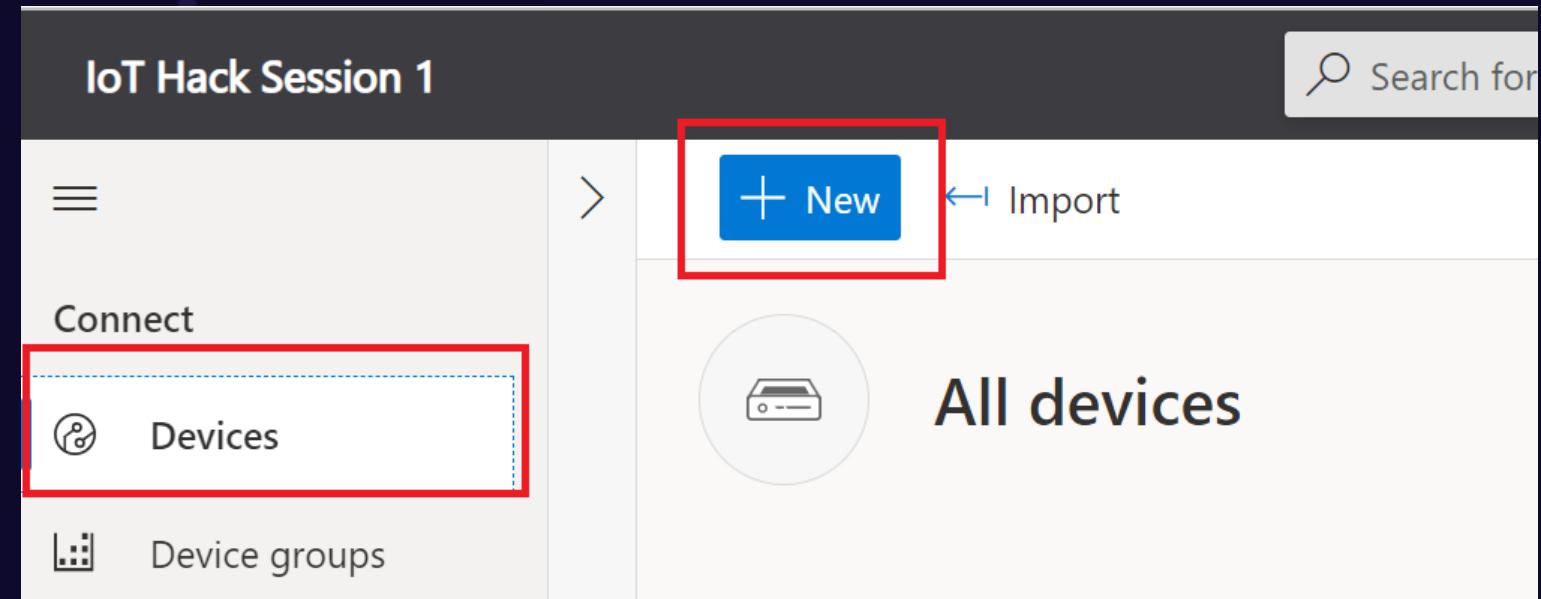
Display name	Name *	Capability type * ⓘ	Semantic type ⓘ
Temperature	Temperature	Telemetry	Temperature
Humidity	Humidity	Telemetry	Humidity

At the bottom left of the table area is a '+ Add capability' button.

This Requires an Azure Subscription!

Try it Out!

Now create an instance of a device based on the model created



This Requires an Azure Subscription!

Try it Out!

Give the device a name

Assigned the previously created device template to this instance of the device

Create the Device

Create a new device

To create a new device, select a device template, a name, and a unique ID. [Learn more](#)

Device name * ⓘ
Samsung Lab Monitor

Device ID * ⓘ
v42xge3m6q

Organization * ⓘ
IoT Hack Session 1

Device template * ⓘ
 Environment Monitor
Unassigned

Before you connect a real device.
 No

Azure IoT Edge device?
Azure IoT Edge moves cloud analytics and custom business logic from the cloud to your devices.
 No

This Requires an Azure Subscription!

Try it Out!

The device has been created in Azure IoT Central.

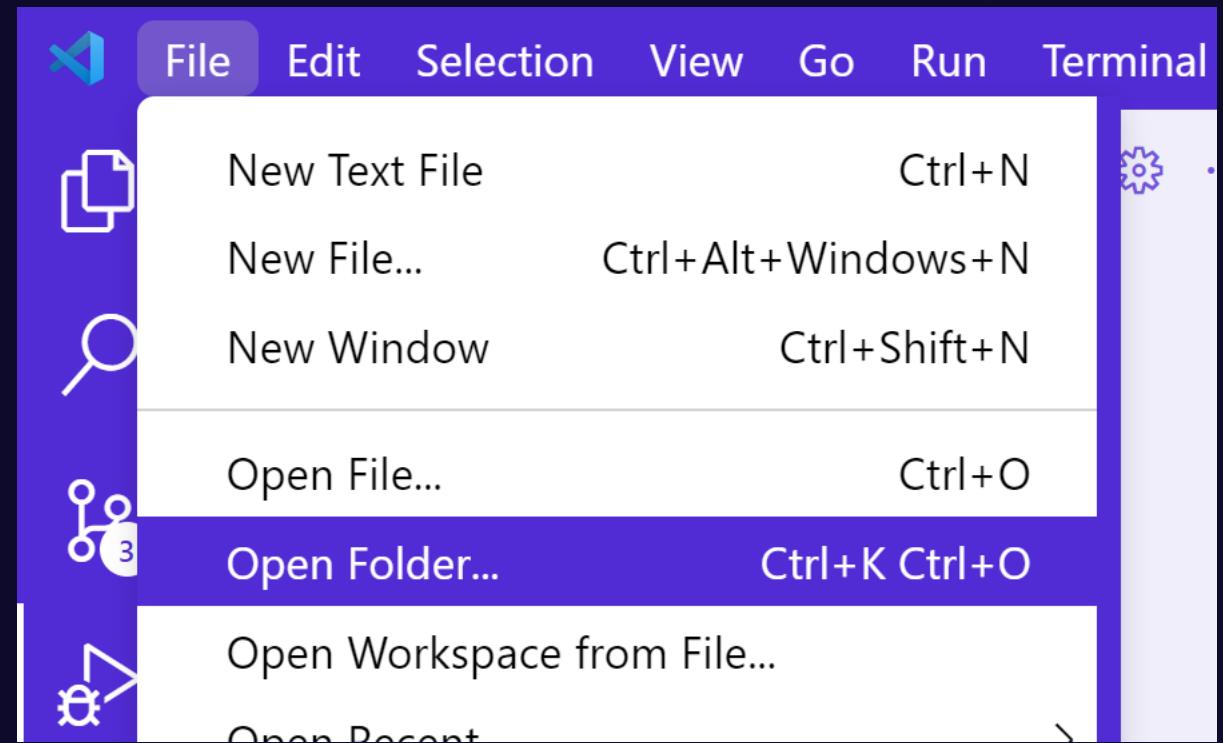
Now lets send it some telemetry from our DHT 11 sensor and Raspberry Pi

The screenshot shows the Azure IoT Central interface for a device named "Samsung Lab Monitor". The left sidebar menu includes sections for Connect (Devices, Device groups, Device templates, Edge manifests), Analyze (Data explorer, Dashboards), Manage (Jobs, Rules, Data export, Security, Audit logs), and Extend. The main content area displays the device's details: "Samsung Lab Monitor", Last data received: N/A, Status: Registered, Organization: IoT Hack Session 1. Below this, tabs for About, Overview, Raw data, Mapped aliases, and Files are shown. The "Overview" tab is selected. Two cards are present: "Temperature, Humidity" and "Temperature". The "Temperature, Humidity" card shows "No data found" and a message: "Check your device or network connection, and make sure you're part of the device's org." The "Temperature" card also shows "No data found" and the same message. The top navigation bar includes "Search for devices", "Connect", "Manage template", "Manage device", and a gear icon.

This Requires an Azure Subscription!

Try it Out!

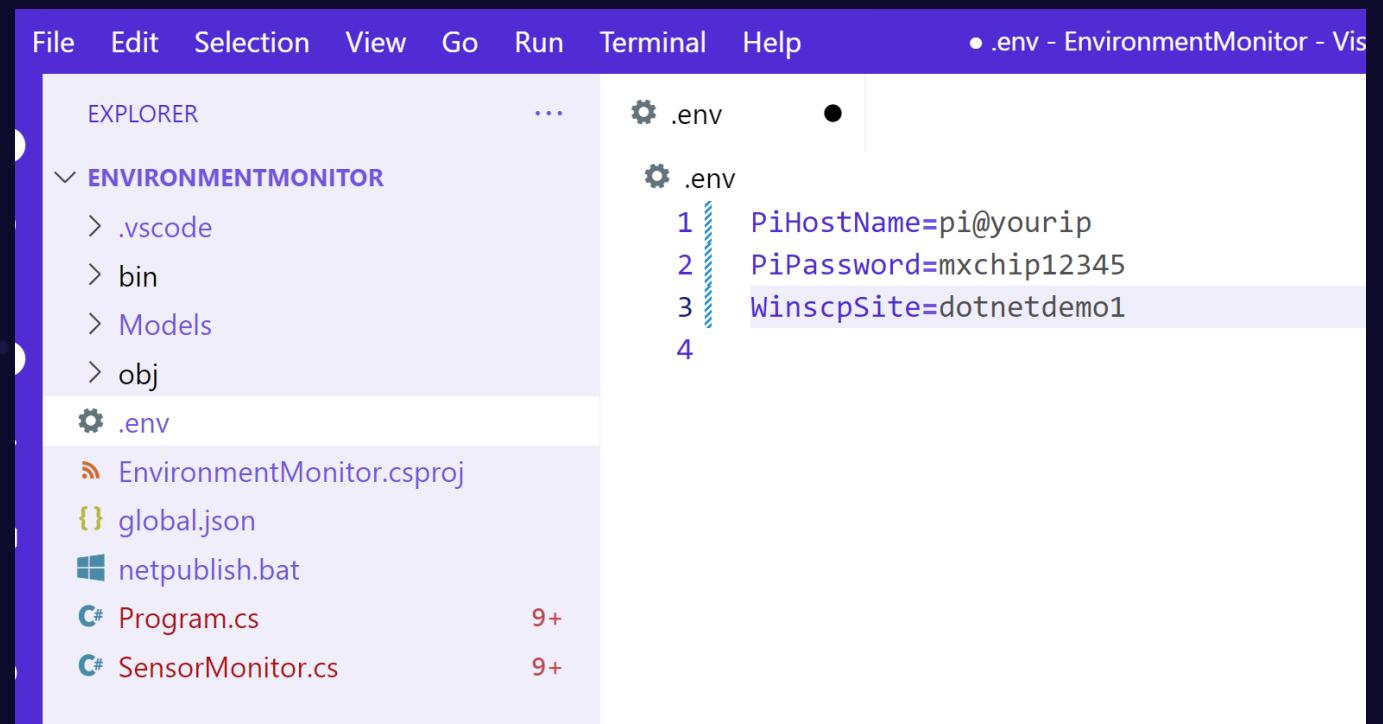
Open the Environment Monitor
folder in Visual Studio Code



This Requires an Azure Subscription!

Try it Out!

Configure the .env file
with your Raspberry Pi's
details



```
File Edit Selection View Go Run Terminal Help • .env - EnvironmentMonitor - Vis

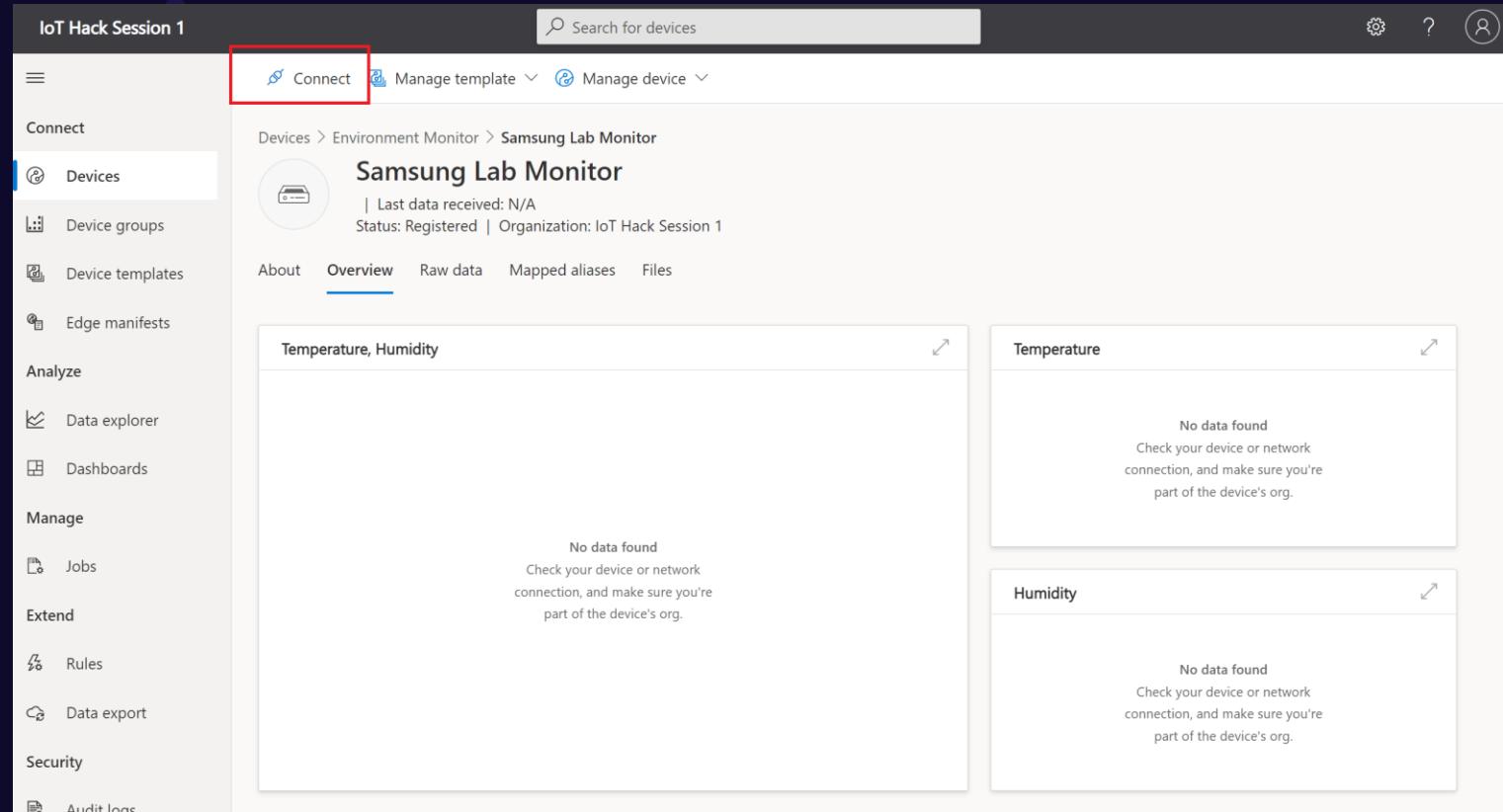
EXPLORER
ENVIRONMENTMONITOR
> .vscode
> bin
> Models
> obj
> .env
> EnvironmentMonitor.csproj
> global.json
> netpublish.bat
C# Program.cs 9+
C# SensorMonitor.cs 9+

.env
1 PiHostName=pi@yourip
2 PiPassword=mxchip12345
3 WinSCP Site=dotnetdemo1
4
```

This Requires an Azure Subscription!

Try it Out!

Within IoT Central select the device and press the “Connect” button.



This Requires an Azure Subscription!

Try it Out!

Take note of the:

ID Scope

Device ID

Primary Key

Device connection groups

ID scope ⓘ

 Copy

Device ID ⓘ

 Copy

Choose the connection type for this device. You can change this later if you need to.

Authentication type

 ▼

Key QR code

Shared Access Signatures (SAS) use security tokens and keys to connect to IoT Central. Use the SAS keys from the default enrollment group shown below to register your device. [Learn more](#)

Primary key ⓘ

 Copy

Secondary key ⓘ

 Copy

This Requires an Azure Subscription!

Try it Out!

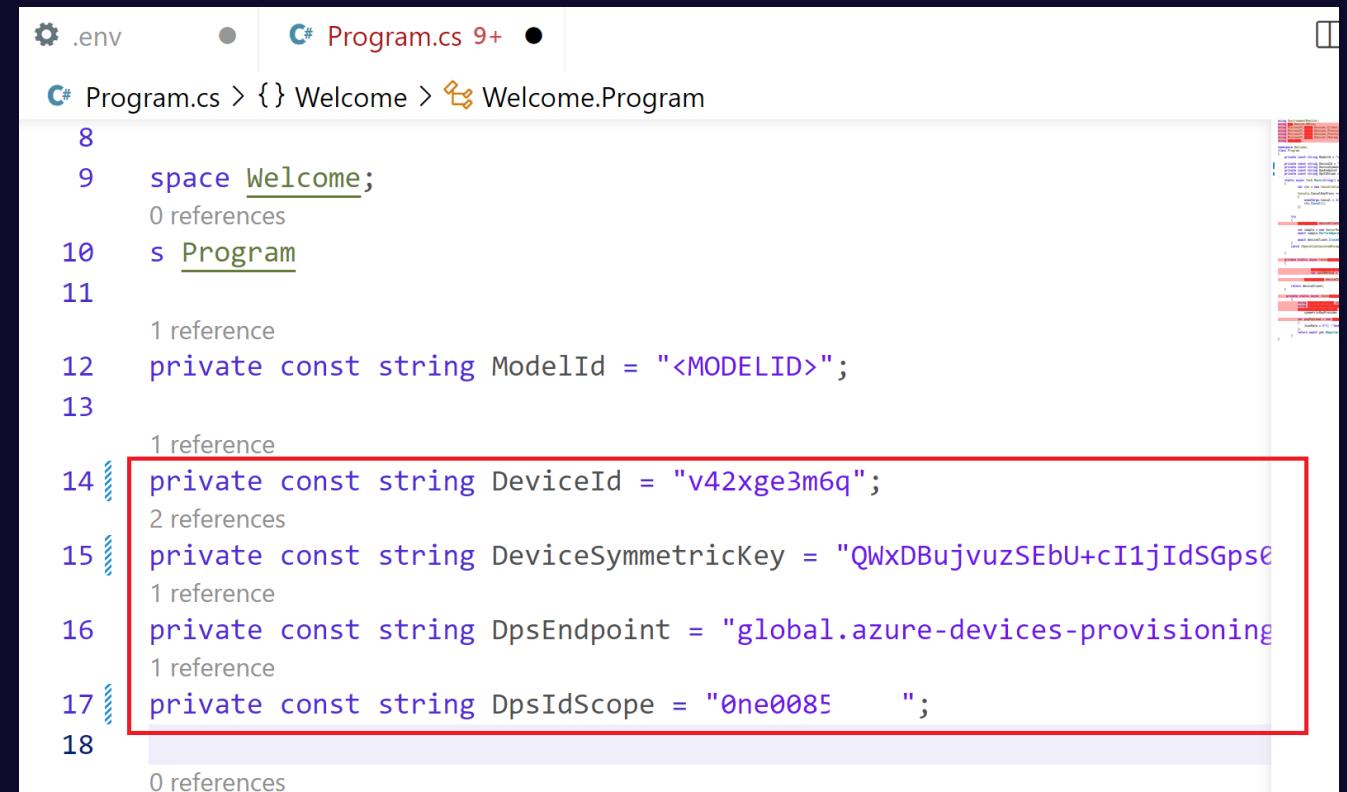
In Visual Studio Code fill in the
Program.cs file

ID Scope -> DpsIdScope

Device ID -> DeviceId

Primary Key ->

DeviceSymmetricKey



```
8
9     space Welcome;
10    s Program
11
12    1 reference
13    private const string ModelId = "<MODELID>";
14    1 reference
15    private const string DeviceId = "v42xge3m6q";
16    2 references
17    private const string DeviceSymmetricKey = "QWxDBuJvuzSEbU+cI1jIdSGpsE";
18    1 reference
19    private const string DpsEndpoint = "global.azure-devices-provisioning";
20    1 reference
21    private const string DpsIdScope = "0ne0085      ";
22
23    0 references
```

This Requires an Azure Subscription!

Try it Out!

Within IoT Central take note of
the

Interface @id

The screenshot shows the Azure IoT Central interface. On the left, a sidebar menu includes 'Connect' (Devices, Device groups, Device templates, Edge manifests), 'Analyze' (Data explorer, Dashboards), 'Manage' (Jobs, Extend), and a 'Device templates' item which is highlighted with a red box. The main content area shows the 'Environment Monitor' device template details. At the top, there are buttons for Version, Manage test device, Publish, Rename, and Delete. Below that, the title 'Environment Monitor' is shown with a status message 'Application updated: 17 minutes ago' and 'Interfaces published: 17 minutes ago'. In the center, there's a table for adding capabilities, with two rows: one for Temperature (Telemetry) and one for Humidity (Telemetry). At the bottom of the central area are Save, Add capability, Edit identity (which is highlighted with a red box), Export, Delete, and more buttons. On the right, a detailed view of the 'Identity' section is shown, including fields for Display name (Environment Monitor), Namespace (iotHackSession1), Name (EnvironmentMonitor_12r), Version (1), and Interface @id (dtmi:iotHackSession1:EnvironmentMonitor;1). The 'Interface @id' field is also highlighted with a red box.

This Requires an Azure Subscription!

Try it Out!

In Visual Studio Code fill in the Programs.cs file

Interface @id -> ModelId

```
8
9  namespace Welcome;
0 references
10 class Program
11 {
12     private const string ModelId = "dtmi:iotHackSession1:Environmental";
13
14     private const string DeviceId = "v42xge3m6q";
15     private const string DeviceSymmetricKey = "QWxDBuJvuzSEbU+cI1jId9";
16     private const string DpsEndpoint = "global.azure-devices-provisioning";
17
18 }
```

This Requires an Azure Subscription!

Try it Out!

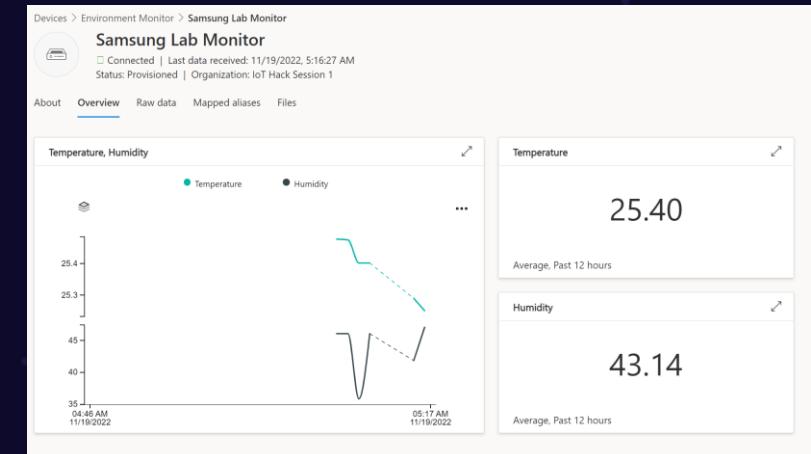
In Visual Studio Code fill in the
Program.cs file

Interface @id -> ModelId

The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Run and Debug Tab:** .NET Remote Launch - Raspberry Pi (highlighted with a red box).
- Code Editor:** Program.cs - EnvironmentMonitor - Visual Studio Code. The code defines a namespace Welcome and a class Program with a private constant ModelId.
- Output Panel:** Shows telemetry data being sent:

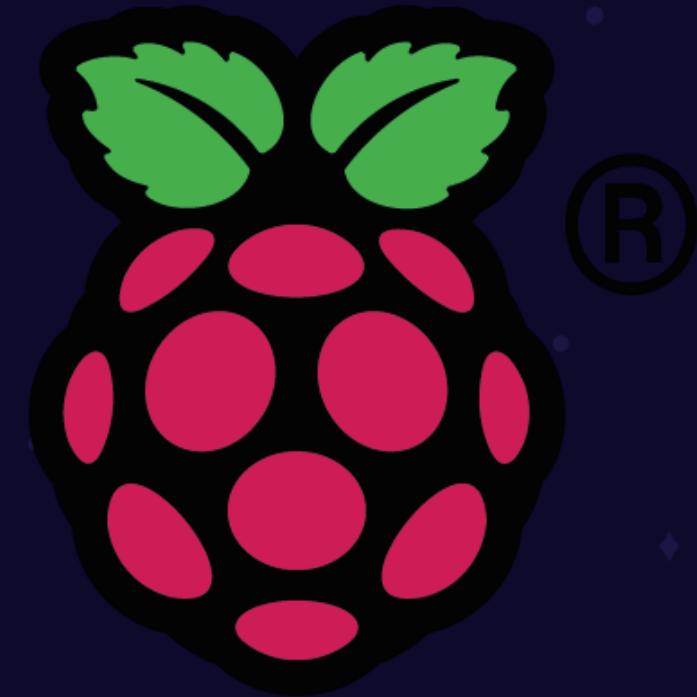
```
Telemetry: Sent - {"Temperature":25.4,"Humidity":46}
Humidity: 46%
Temperature: 25.4C
Telemetry: Sent - {"Temperature":25.4,"Humidity":46}
Humidity: 46%
Temperature: 25.4C
Telemetry: Sent - {"Temperature":25.4,"Humidity":46}
```



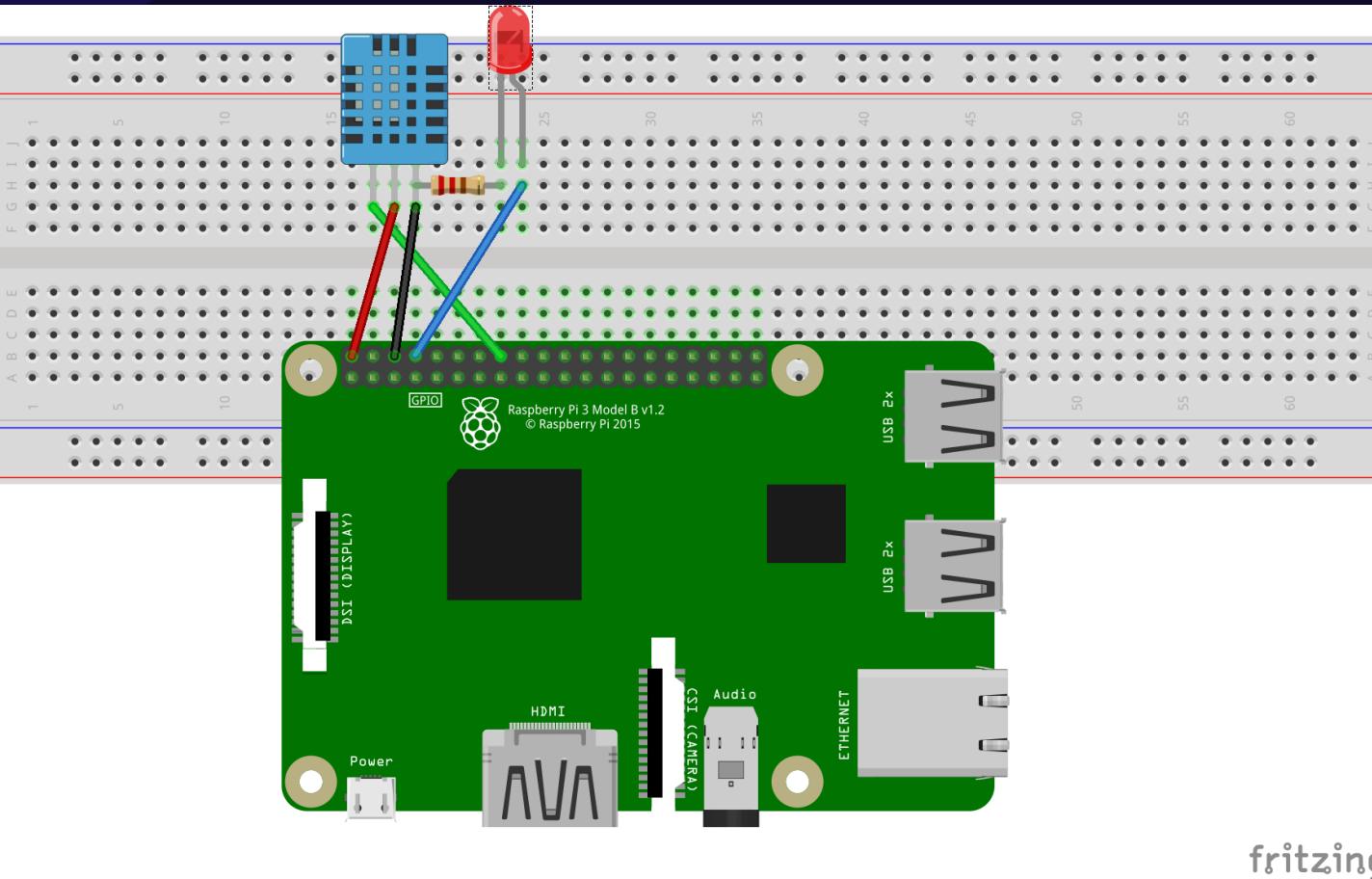
This Requires an Azure Subscription!

Module 4

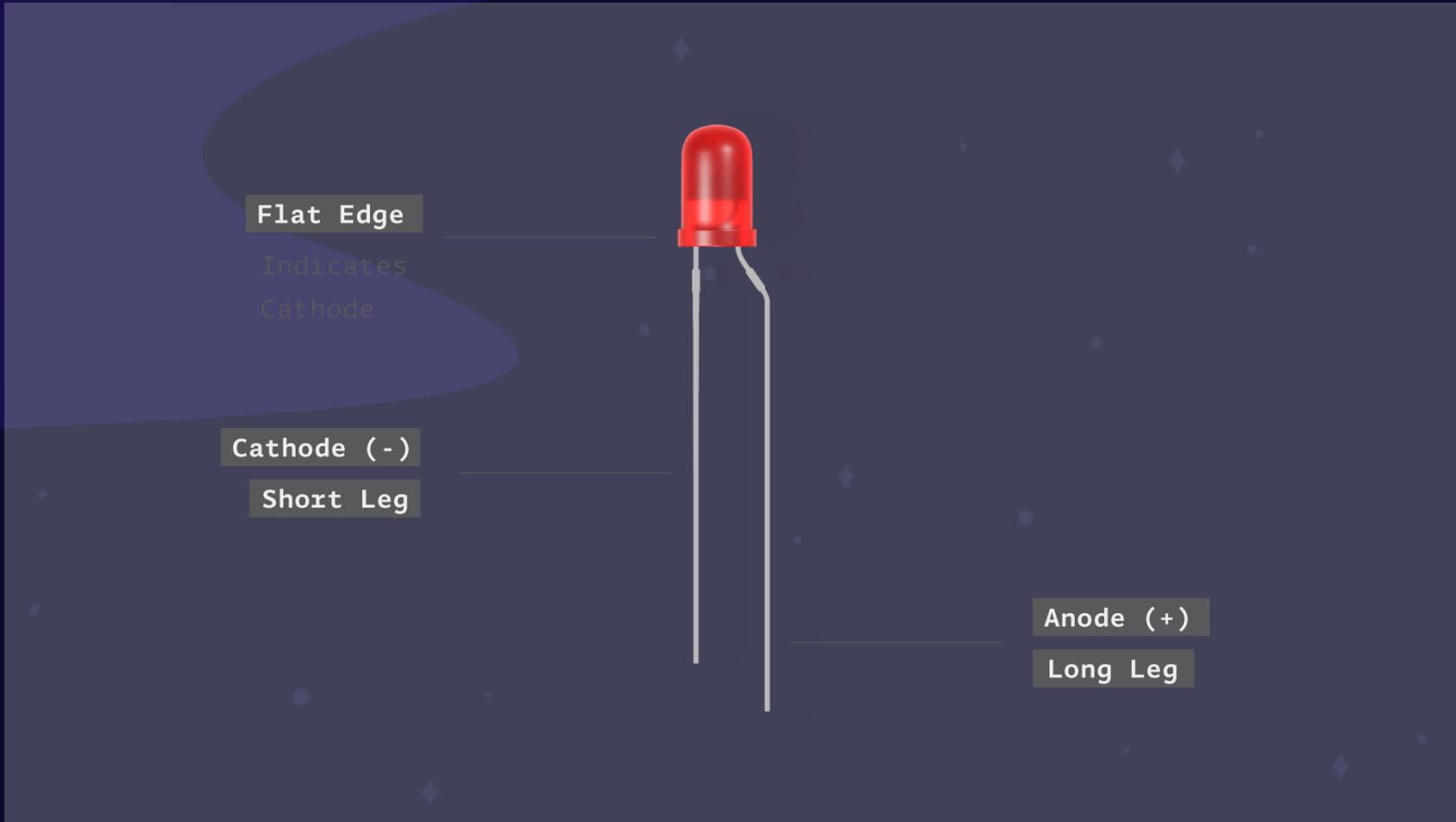
LED Fun



IoT is not IoT without a LED!



IoT is not IoT without a LED!



IoT is not IoT without a LED!

Why the resistor?

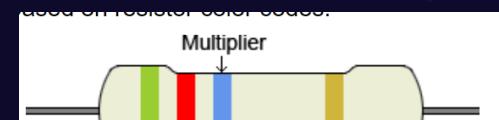
In this circuit the resistor is used to reduce current flow.

The Raspberry Pi is designed to only provide 23 mA max current. A resistor is important to not damage both the Raspberry Pi and the LED

<https://www.calculator.net/resistor-calculator.html>

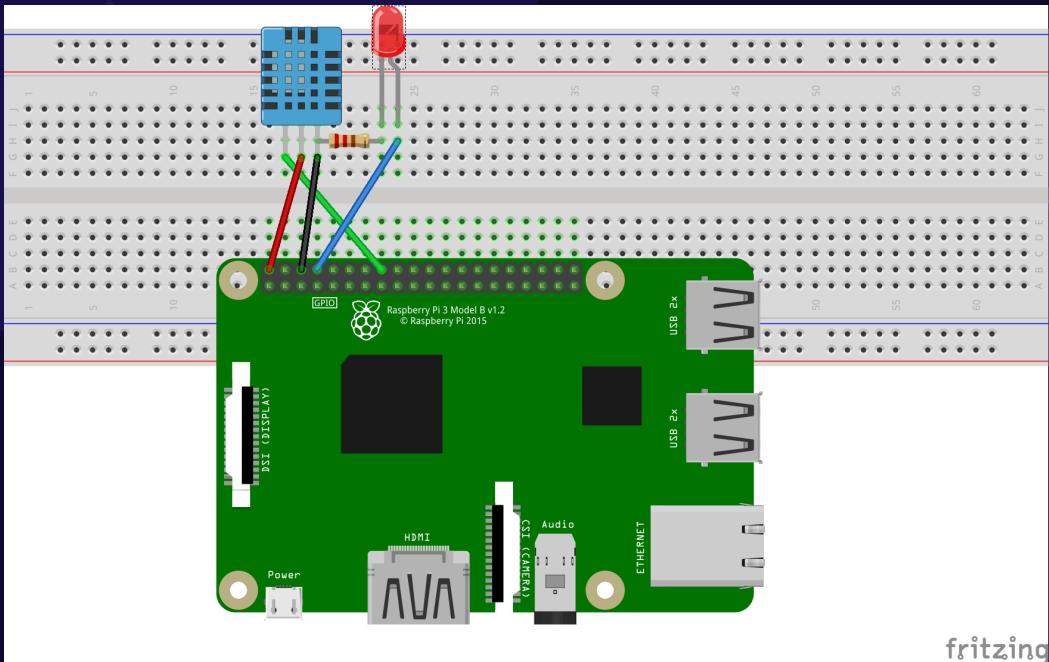
$$(V_{\text{supply}} - V_F) / I_F = (3.3 - 1.7) / 7\text{mA} = 220 \Omega$$

Based on resistor color codes.



Color	1 st , 2 nd Band Significant Figures	Multiplier	Tolerance
Black	0	$\times 1$	
Brown	1	$\times 10$	$\pm 1\% (\text{F})$
Red	2	$\times 100$	$\pm 2\% (\text{G})$
Orange	3	$\times 1\text{K}$	$\pm 0.05\% (\text{W})$
Yellow	4	$\times 10\text{K}$	$\pm 0.02\% (\text{P})$
Green	5	$\times 100\text{K}$	$\pm 0.5\% (\text{D})$
Blue	6	$\times 1\text{M}$	$\pm 0.25\% (\text{C})$
Violet	7	$\times 10\text{M}$	$\pm 0.1\% (\text{B})$
Grey	8	$\times 100\text{M}$	$\pm 0.01\% (\text{L})$
White	9	$\times 1\text{G}$	
Gold		$\times 0.1$	$\pm 5\% (\text{J})$
Silver		$\times 0.01$	$\pm 10\% (\text{K})$

IoT is not IoT without a LED!



Using the component pack, build the circuit using your Raspberry Pi, breadboard, LED, resistor and wires.

Connect resistor from ground (on DHT11) to LED cathode (short leg)

GPIO Pin 14 from the Raspberry Pi to LED anode

Try it out!

Coding Challenge

Turn the LED on when a certain temperature is reached. (check ambient temperature and use that as a starting point)

Hints

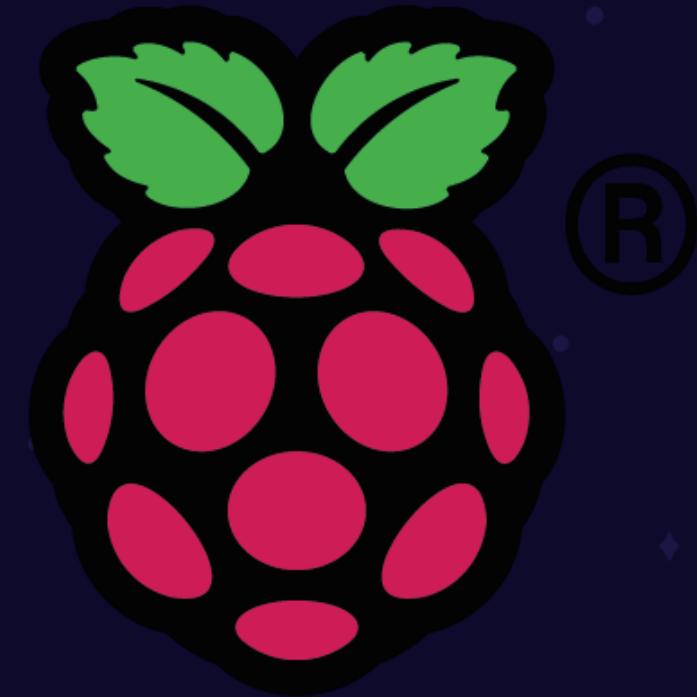
```
int ledPin = 14;  
  
var controller = new GpioController();  
  
controller.OpenPin(ledPin,PinMode.Output)  
  
controller.Write(ledPin,PinValue.High);
```

<https://learn.microsoft.com/en-us/dotnet/api/system.device.gpio.gpiocontroller?view=iot-dotnet-latest>

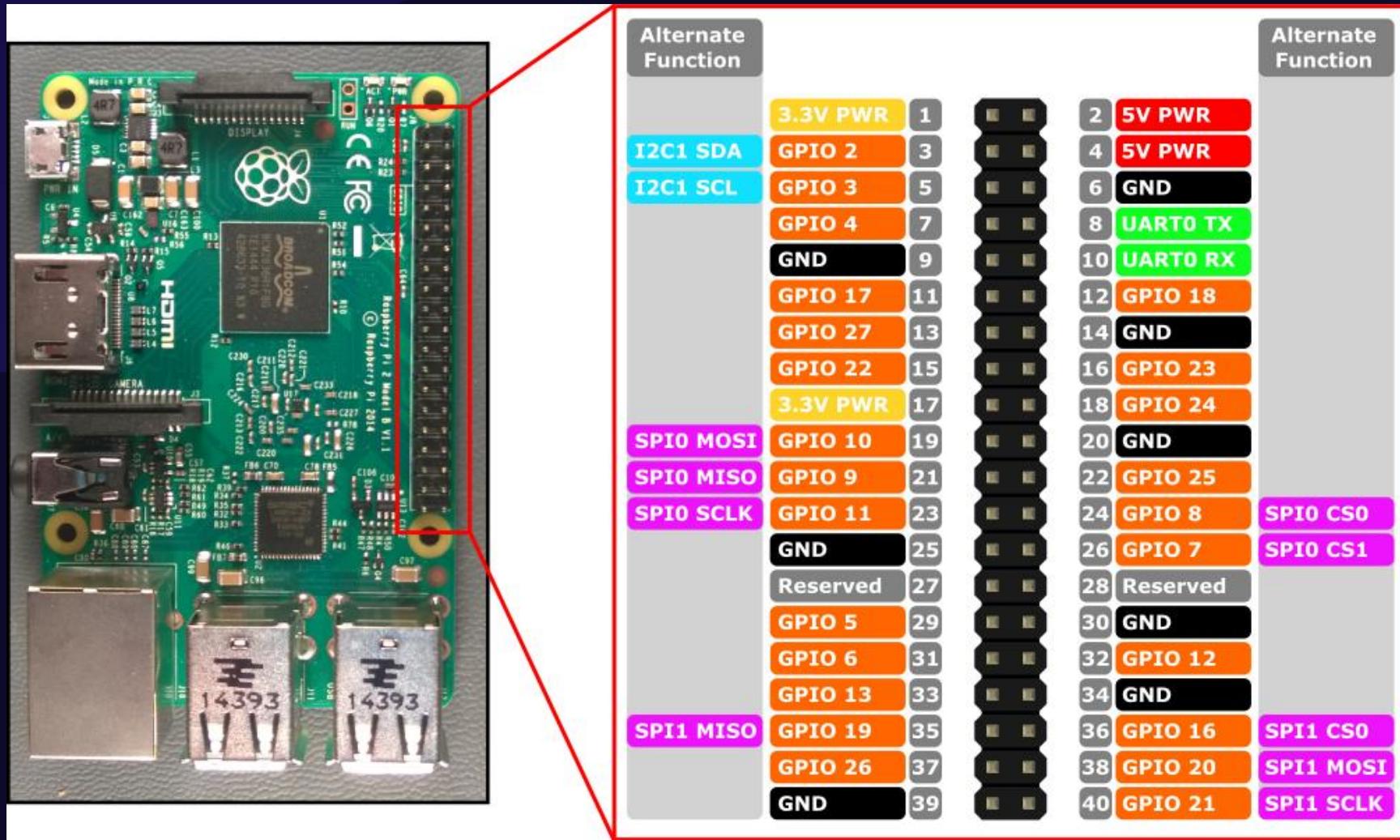
System.Device.Gpio

Module 5

Raspberry Pi Hardware + HATS



Interfacing to the Raspberry Pi



Interfacing to the Raspberry Pi

GPIO - General-purpose input/output

Switching things on, switching things off

Analog

No just 1 (on) and 0 (off)

Variable Input and Output voltage.

Raspberry Pi has no Analog pins. Requires an Analog to Digital Converter

I2C

Serial communication Bus

Handy as it's usually 2 wires for complex data communication

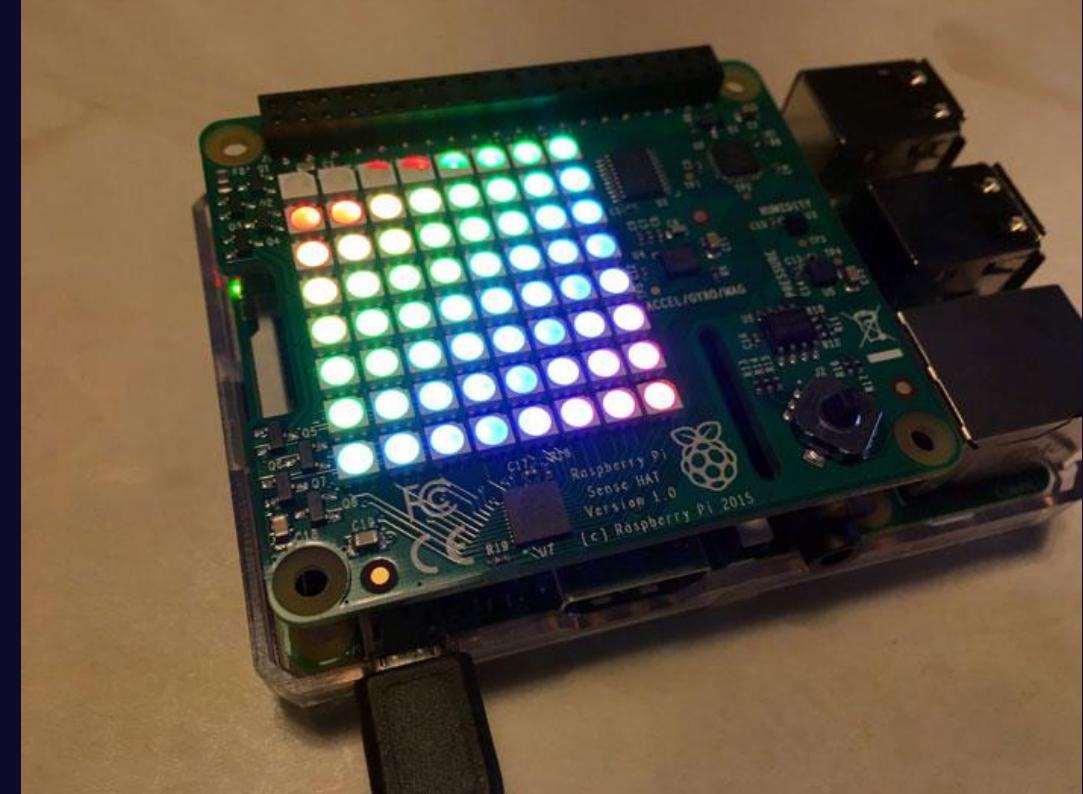
SPI (Serial Peripheral Interface)

Serial communication bus

Usually 3 wire communication

What is a HaT?

- Hardware Attached on Top
- Add-on board
- Fits onto the GPIO pins
- Input/output ports, sensors, displays, motor controllers
- Plug-and-play



Enviro for Raspberry Pi

- pHAT
- BME280 temperature, pressure, humidity sensor
- LTR-559 light and proximity sensor
- MICS6814 analog gas sensor
- ADS1015 analog to digital converter (ADC)
- MEMS microphone
- 0.96" colour LCD (160x80)
- Particulate matter (PM) sensor



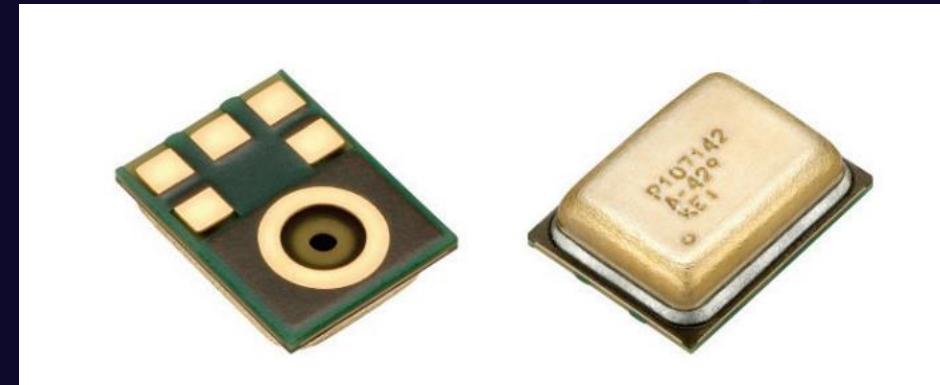
BME 280

- Pressure
- Humidity
- Air Pressure
- I2C or SPI interface
- Weather monitoring, indoor climate control, and industrial automation



MEMS Microphone

- Electro-acoustic transducer housing a sensor
- High SNR of 65dB(A)
- Low Current of typ. 600µA
- I2C interface



Small portable devices: wearables

Set-top boxes: TV, gaming, remote controllers

Smart home devices, Internet of Things,
Connected equipment

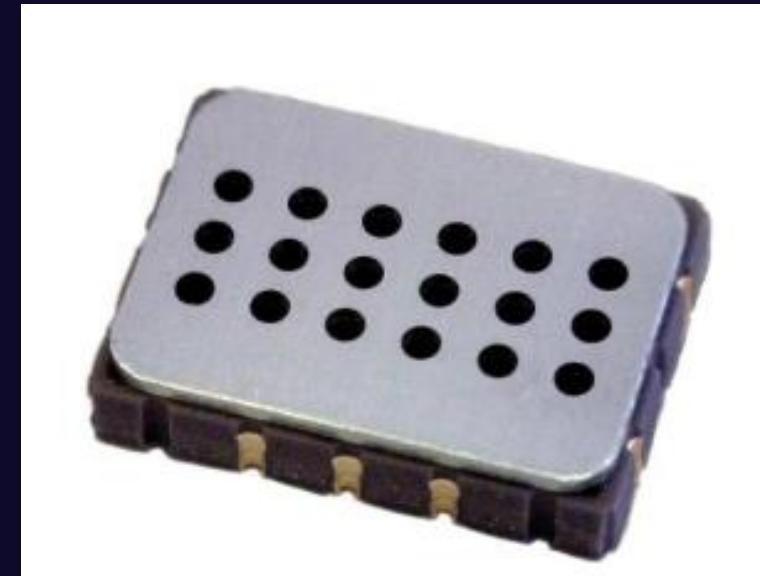
Particulate Matter Sensor

- Monitor air pollution
- PM1, PM2.5, PM10
 - Respiratory
 - Cardiovascular problems in humans
- Smoke, dust, pollen, metal and organic particles
- Fan that sucks air through the sensor and past a laser
- Detect number / concentration and size of particles



MICS6814

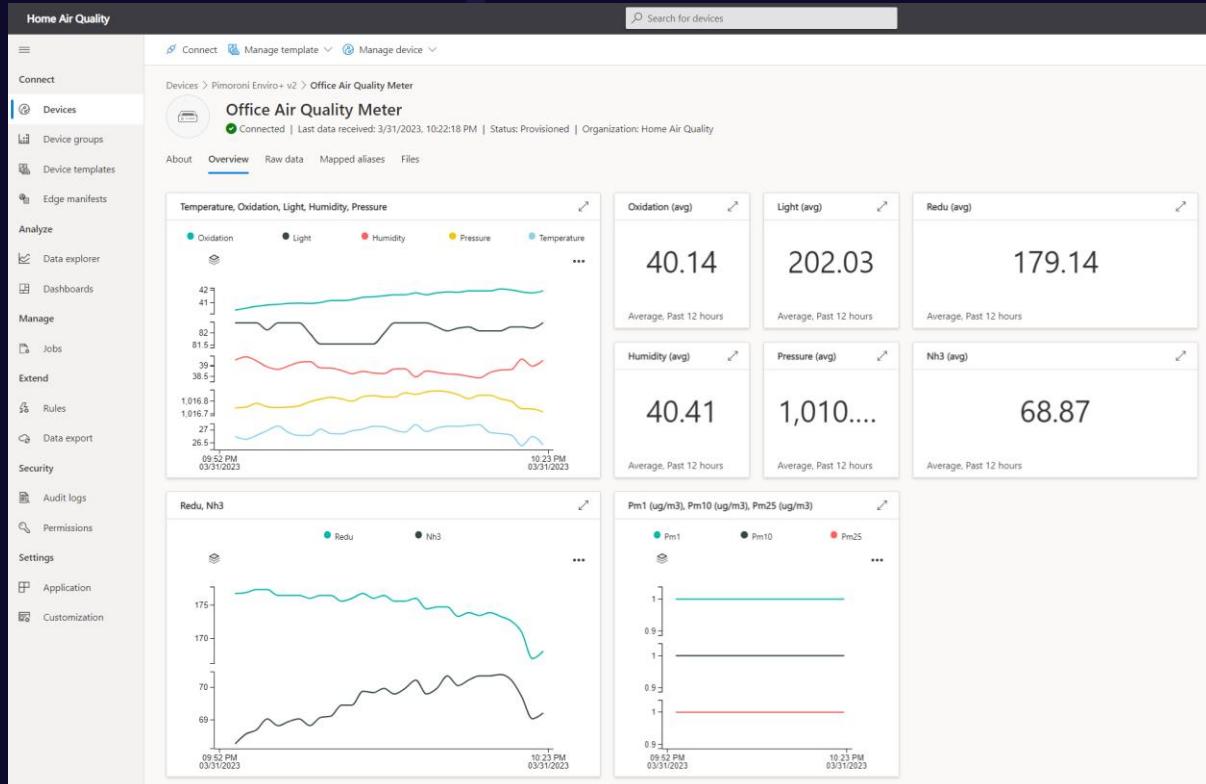
- Analog gas sensor
- MEMS sensor
- Detectable gases
 - Carbon monoxide CO 1 – 1000ppm
 - Nitrogen dioxide NO₂ 0.05 – 10ppm
 - Ethanol C₂H₅OH 10 – 500ppm
 - Hydrogen H₂ 1 – 1000ppm
 - Ammonia NH₃ 1 – 500ppm
 - Methane CH₄ >1000ppm
 - Propane C₃H₈ >1000ppm
 - Iso-butane C₄H₁₀ >1000ppm
- Detection of pollution from automobile exhausts
- Agricultural / industrial odours
- Dangerous gases in the environment



Enviro for Raspberry Pi in Action

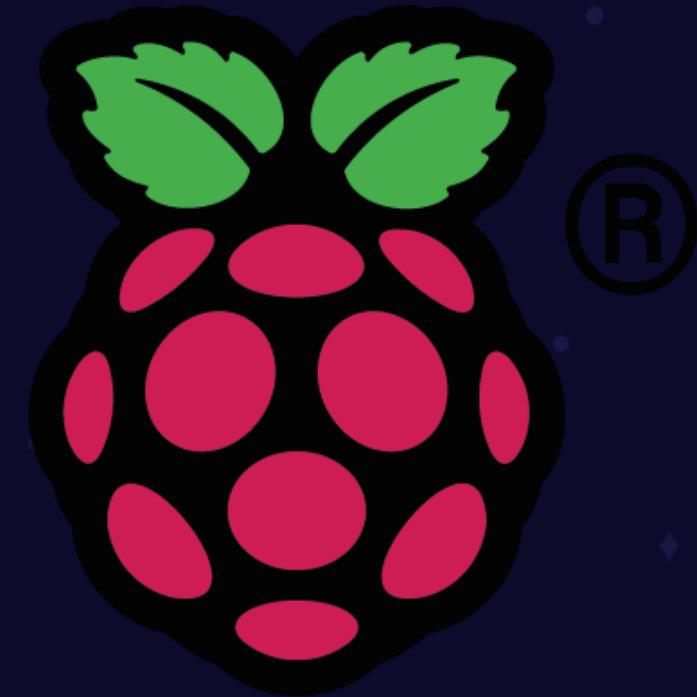
```
2023-03-31 21:36:34.192 INFO    Displays readings from all of Enviro plus' sensors
Press Ctrl+C to exit!

- [onconnect] => status:0
  - [onmessage:sent] =>
2023-03-31 21:36:43.988 INFO    temp: 14.6 C
2023-03-31 21:36:43.998 INFO    pres: 673.0 hPa
2023-03-31 21:36:44.009 INFO    humi: 43.1 %
2023-03-31 21:36:44.025 INFO    ligh: 82.1 Lux
2023-03-31 21:36:44.133 INFO    oxid: 453.1 kO
2023-03-31 21:36:44.136 INFO    redu: 117.5 kO
2023-03-31 21:36:44.139 INFO    nh3: 21.5 kO
2023-03-31 21:36:44.145 INFO    pm1: 0.0 ug/m3
2023-03-31 21:36:44.149 INFO    pm25: 0.0 ug/m3
2023-03-31 21:36:44.153 INFO    pm10: 0.0 ug/m3
Sending telemetry..
2023-03-31 21:37:04.257 INFO    temp: 24.4 C
2023-03-31 21:37:04.268 INFO    pres: 1016.7 hPa
2023-03-31 21:37:04.280 INFO    humi: 43.1 %
2023-03-31 21:37:04.292 INFO    ligh: 82.1 Lux
2023-03-31 21:37:04.359 INFO    oxid: 20.7 kO
2023-03-31 21:37:04.362 INFO    redu: 192.4 kO
2023-03-31 21:37:04.365 INFO    nh3: 48.4 kO
2023-03-31 21:37:04.372 INFO    pm1: 0.0 ug/m3
2023-03-31 21:37:04.375 INFO    pm25: 1.0 ug/m3
2023-03-31 21:37:04.379 INFO    pm10: 1.0 ug/m3
Sending telemetry..
```



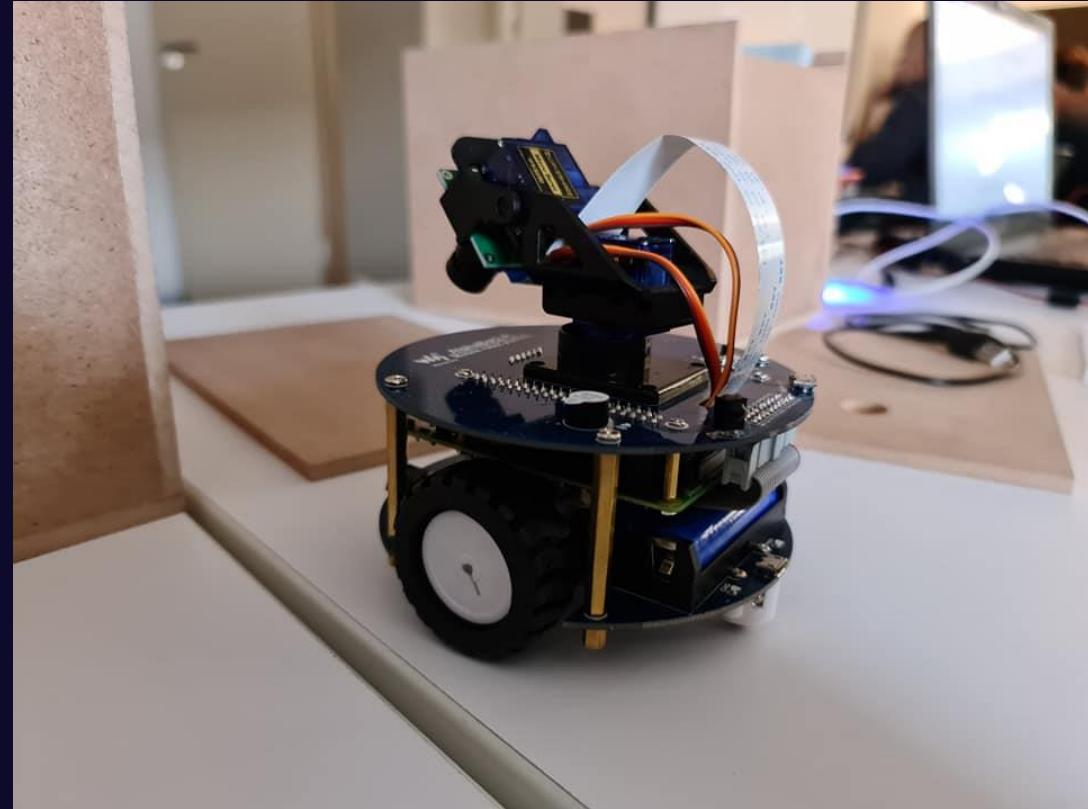
Module 6

The Alphabot 2



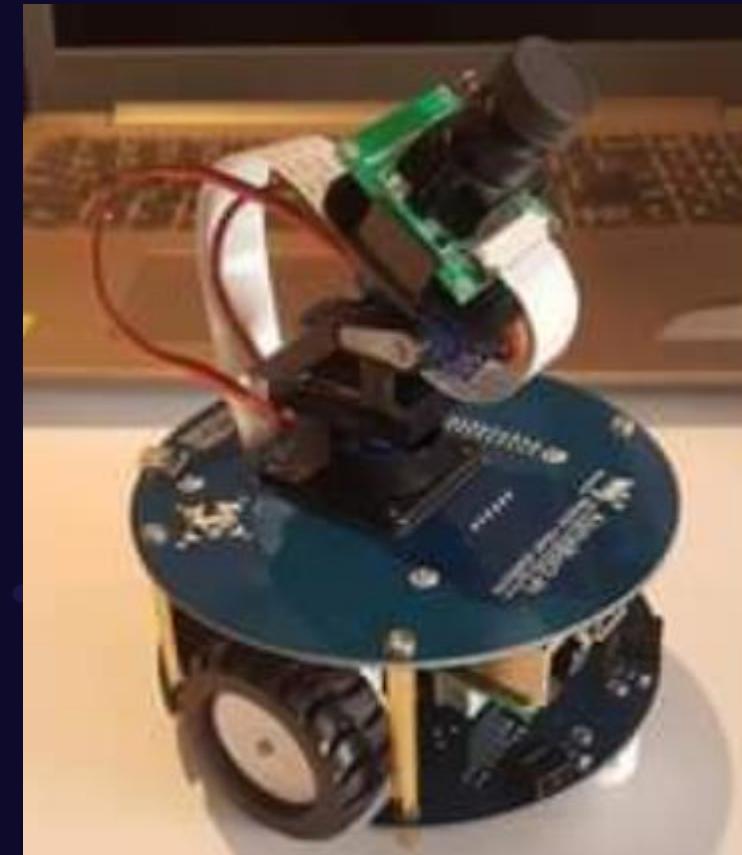
Module 6

The Alphabot 2



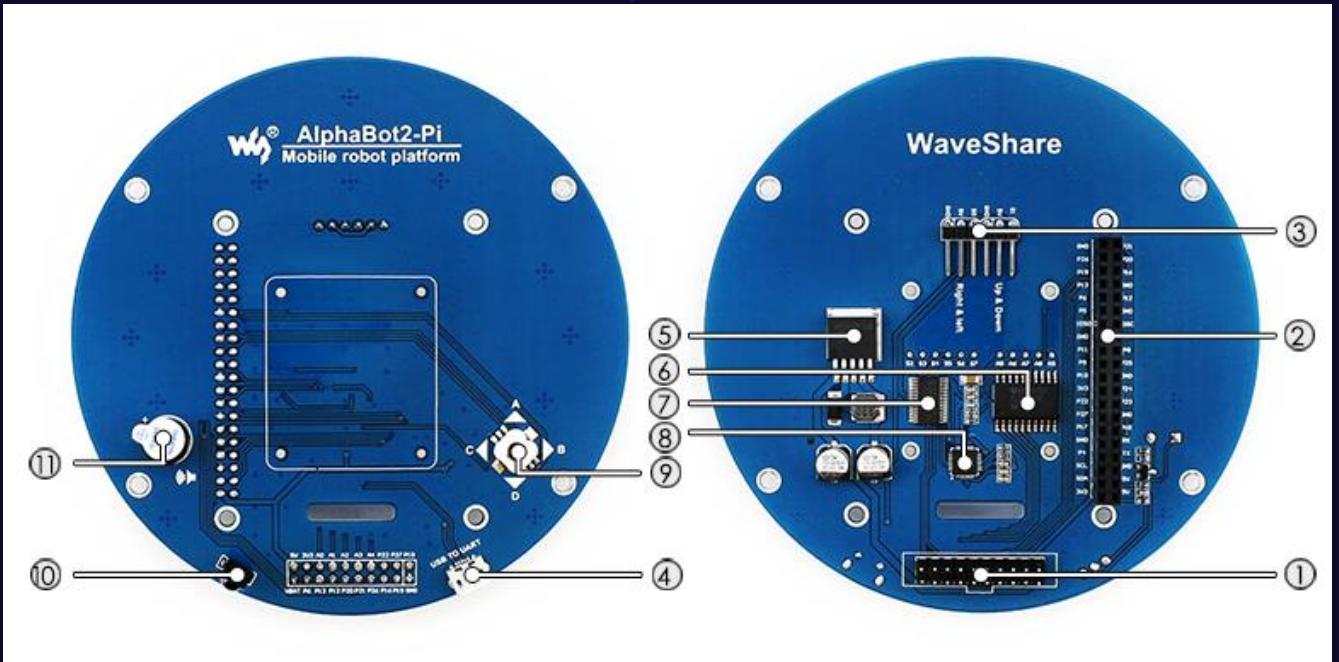
What is an Alphabot 2?

- HaT for a Raspberry Pi
- Robot Platform
- Supports Raspberry Pi 3 + 4



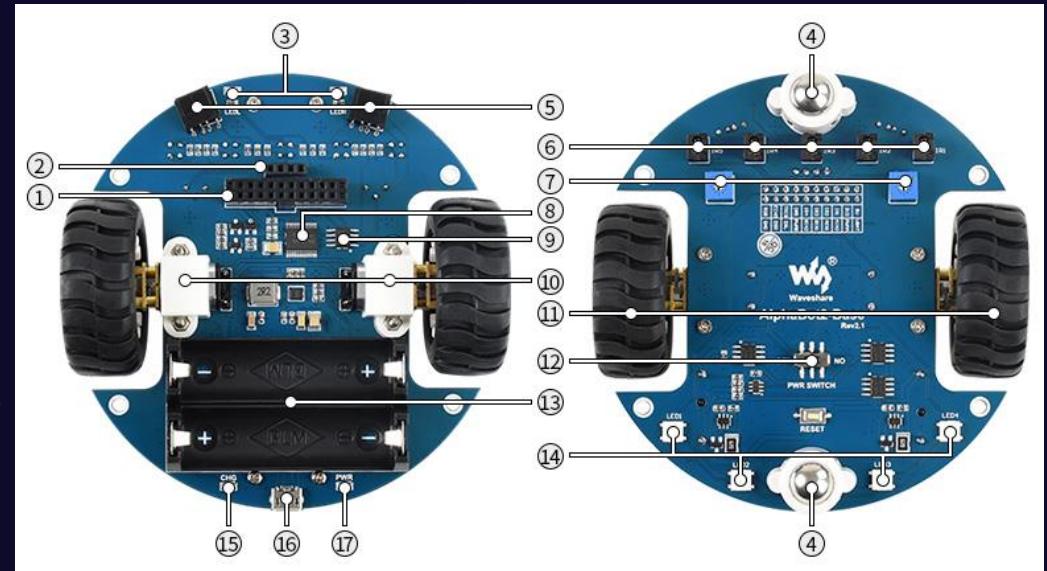
What's in it?

1. AlphaBot2 control interface: for connecting AlphaBot2-Base
2. Raspberry Pi interface: for connecting Raspberry Pi 3 / 4 Model B
3. Servo interface
4. USB TO UART: easy for controlling the Pi via UART
5. LM2596: 5V voltage regulator
6. TLC1543: 10-bit AD acquisition chip, allows the Pi to use analog sensors
7. PCA9685: servo controller, make it more smoothly to rotate the pan head
8. CP2102: USB TO UART converter
9. Joystick
10. IR receiver
11. Buzzer



What's in it?

1. AlphaBot2 control interface: for connecting sorts of the controller adapter board
2. Ultrasonic module interface
3. Obstacle avoiding indicators
4. Omni-direction wheel
5. ST188: reflective infrared photoelectric sensor, for obstacle avoiding
6. ITR20001/T: reflective infrared photoelectric sensor, for line tracking
7. Potentiometer for adjusting obstacle avoiding range
8. TB6612FNG dual H-bridge motor driver
9. LM393 voltage comparator
10. N20 micro gear motor reduction rate 1:30, 6V/600RPM
11. Rubber wheels diameter 42mm, width 19mm
12. Power switch
13. Battery holder: supports 14500 batteries
14. WS2812B: true-color RGB LEDs
15. Battery charging indicator
16. 5V USB battery charging port
17. Power indicator



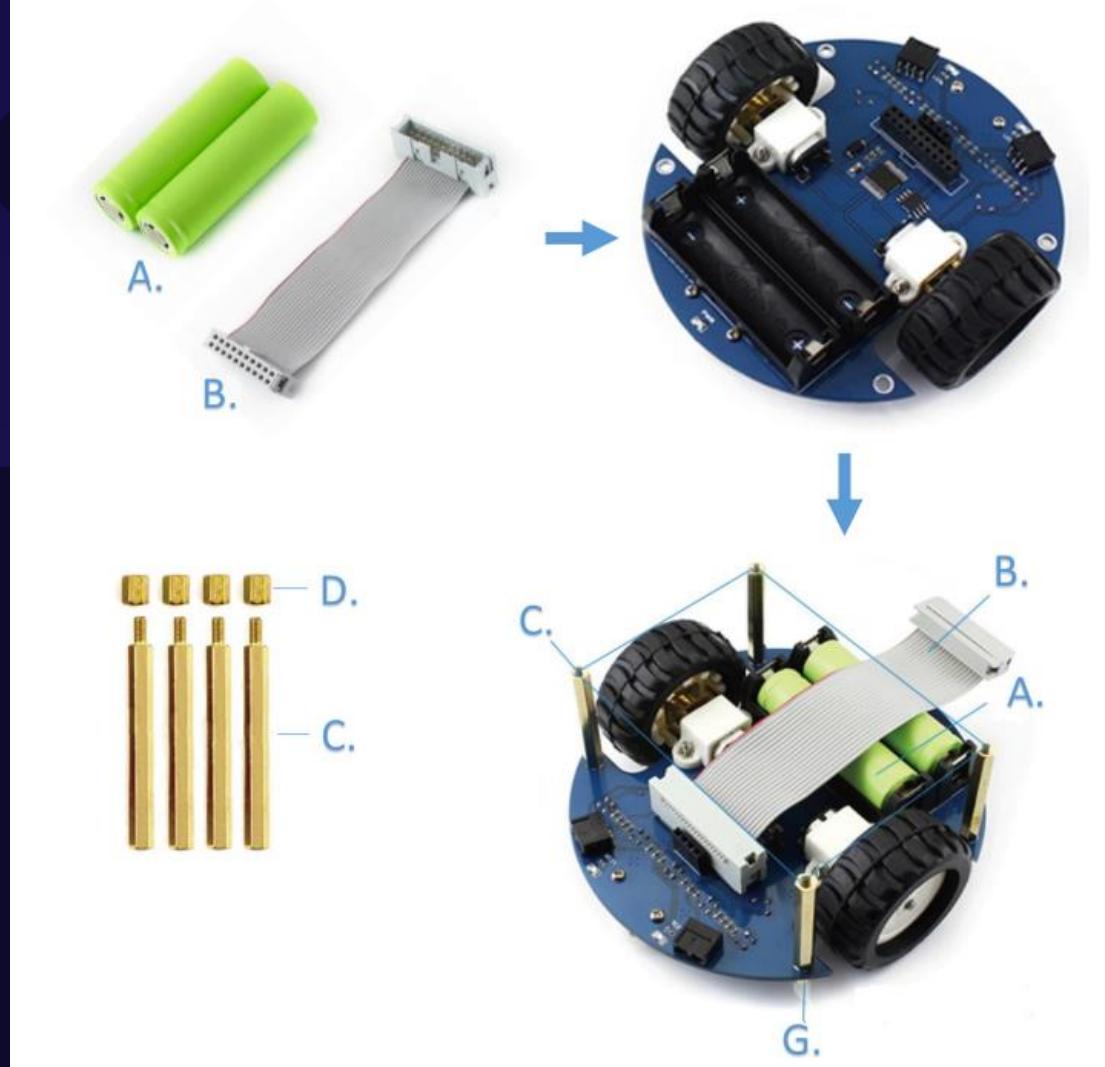
Let's Build a Bot

Before you start. Write down
your host name!!!

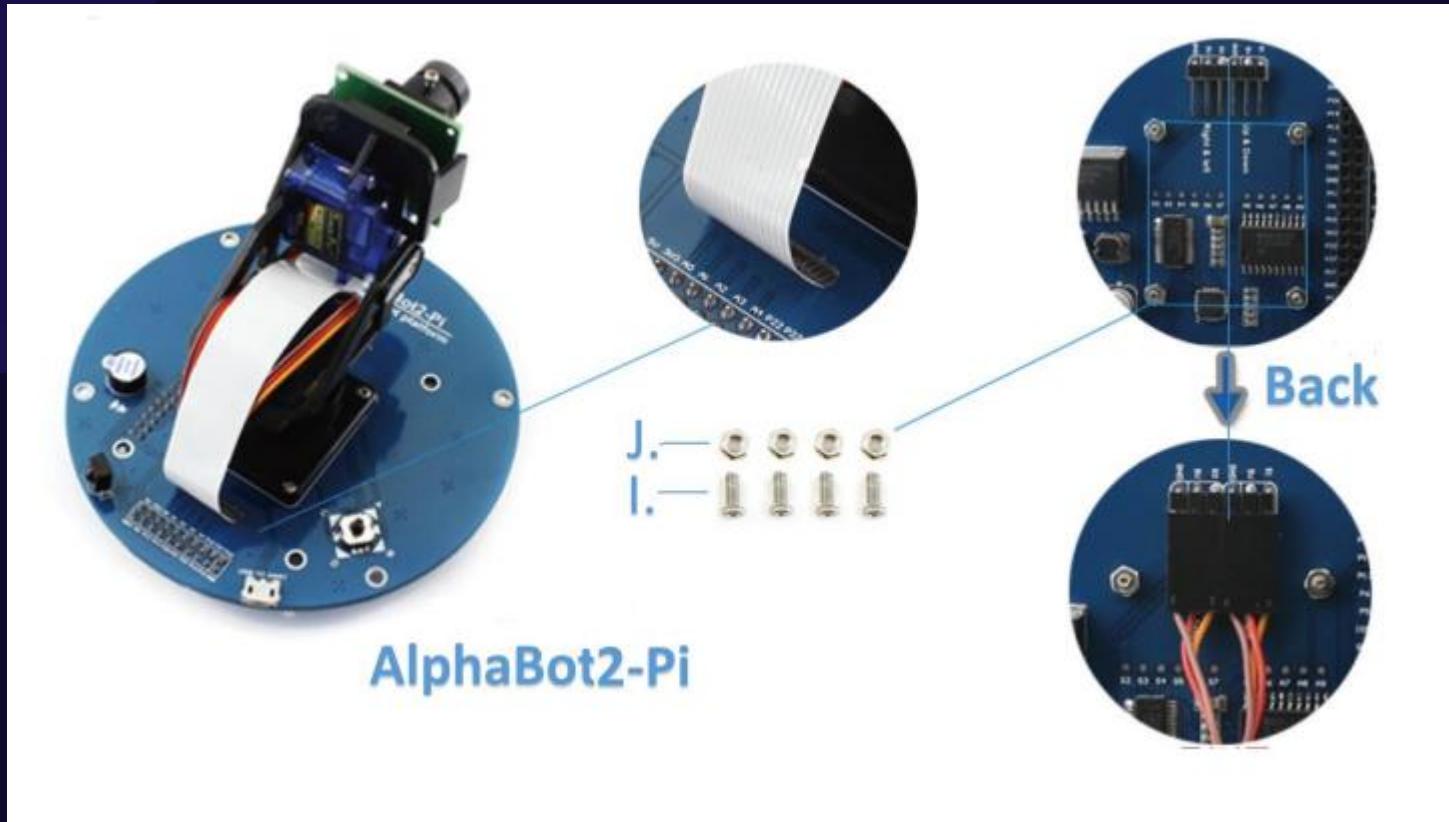
The sticker on your Pi.



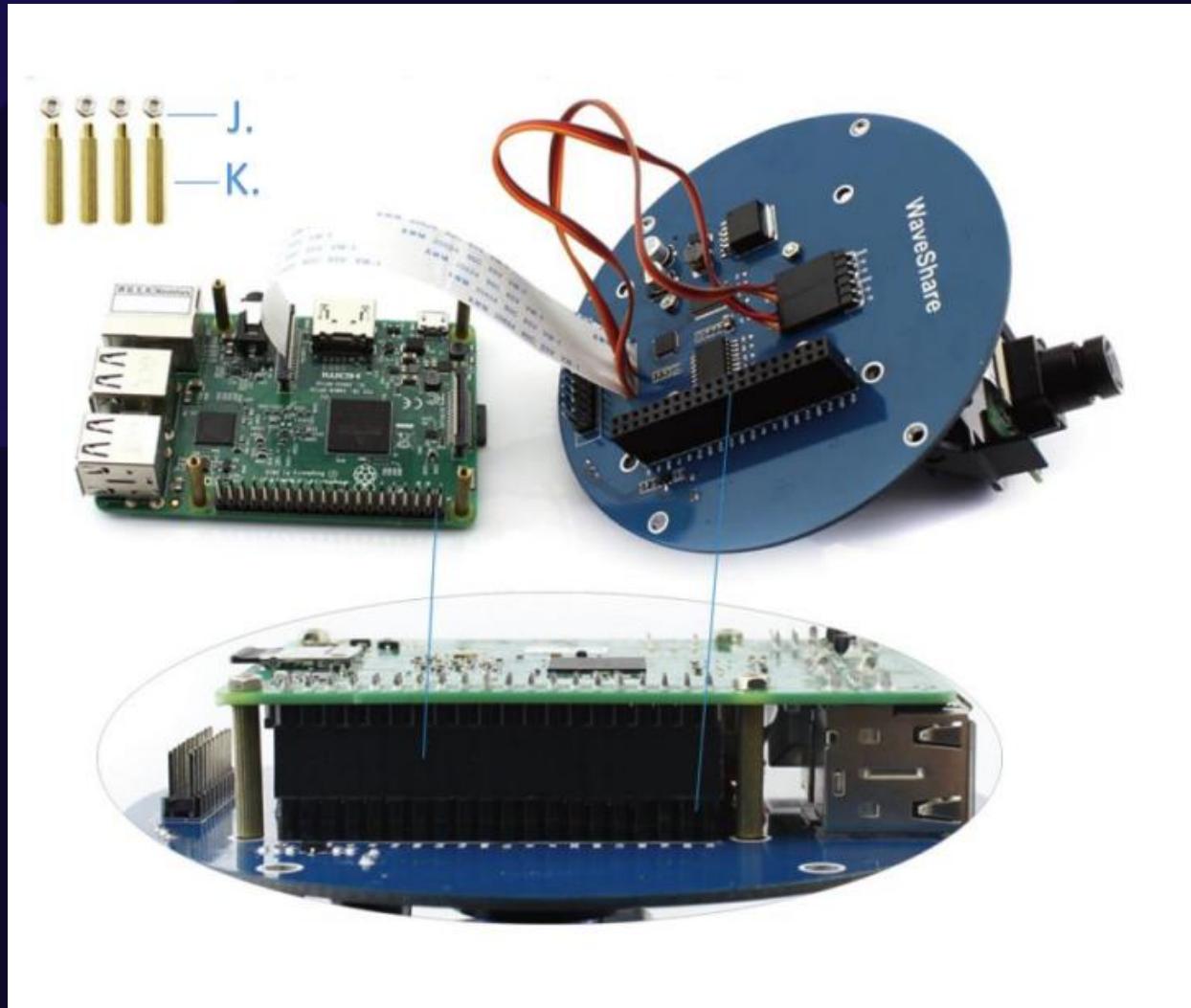
Let's Build a Bot



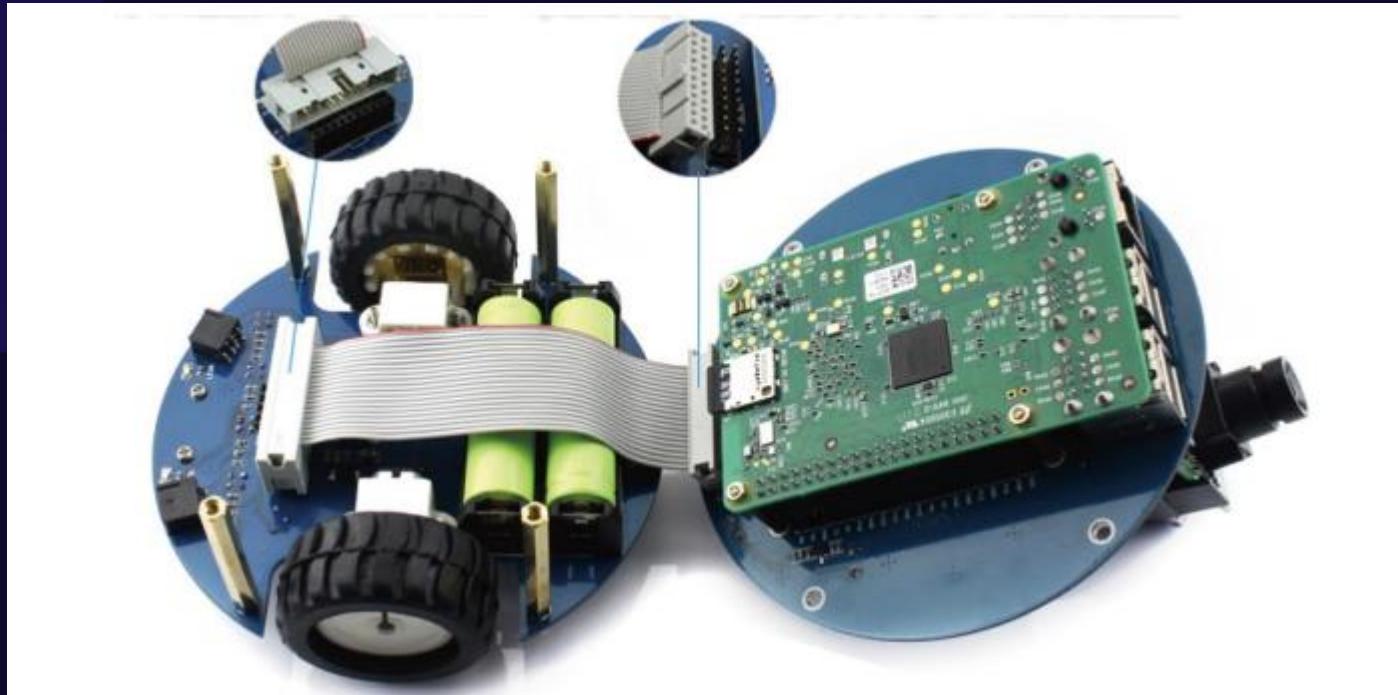
Let's Build a Bot



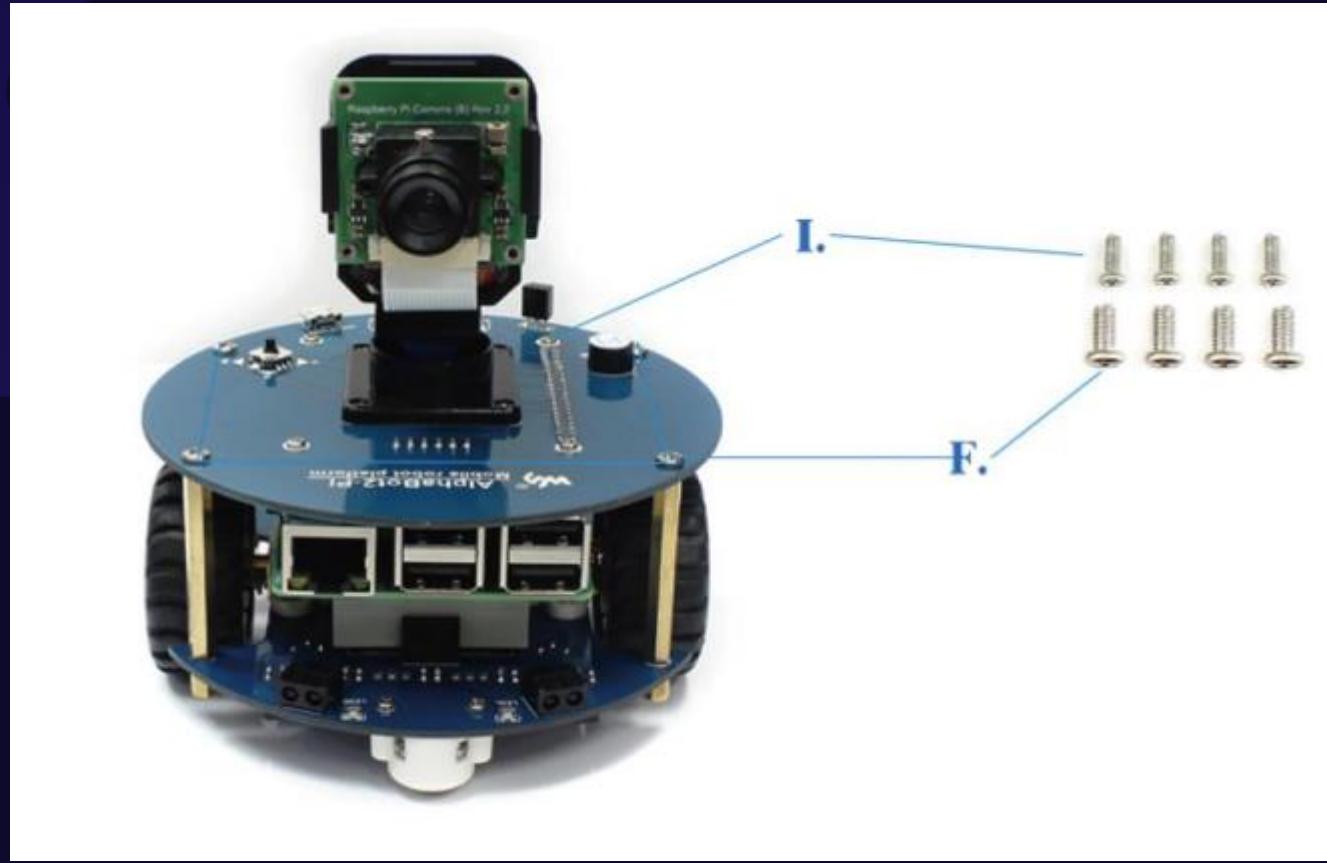
Let's Build a Bot



Let's Build a Bot

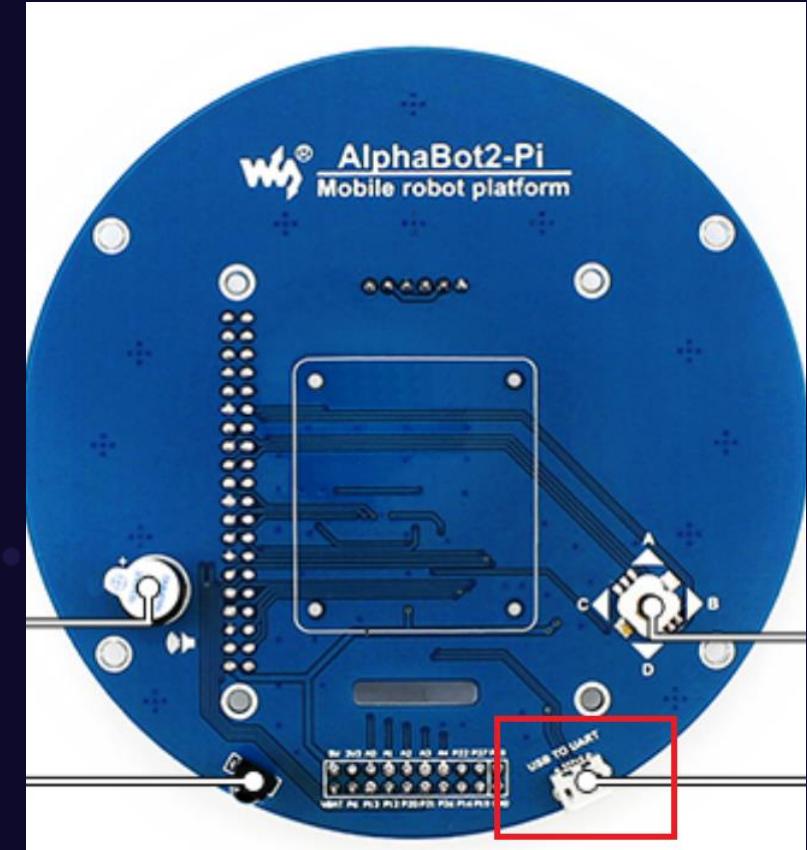


Let's Build a Bot



Plug your Bot in

- Plug in Ethernet network cable
- You can power the Bot with the power supplies provided or a USB cable in the UART port.



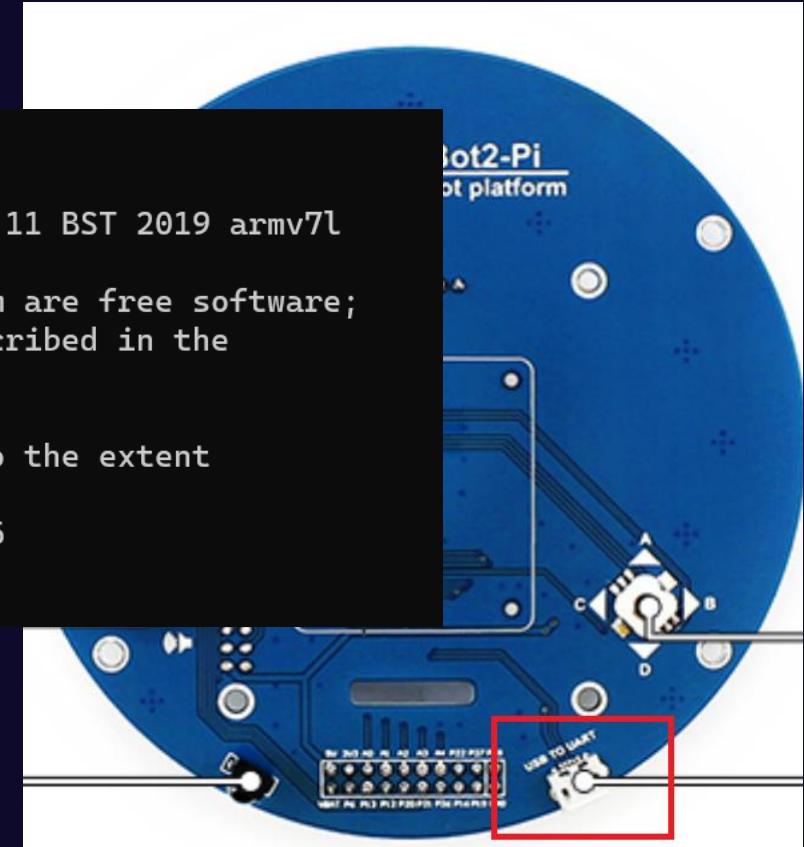
Check if your bot is alive!

- SSH to
- User Na
- Passwo

```
C:\Events\iothackdays2wip>ssh pi@E1163153
pi@e1163153's password:
Linux E1163153 4.19.75-v7+ #1270 SMP Tue Sep 24 18:45:11 BST 2019 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

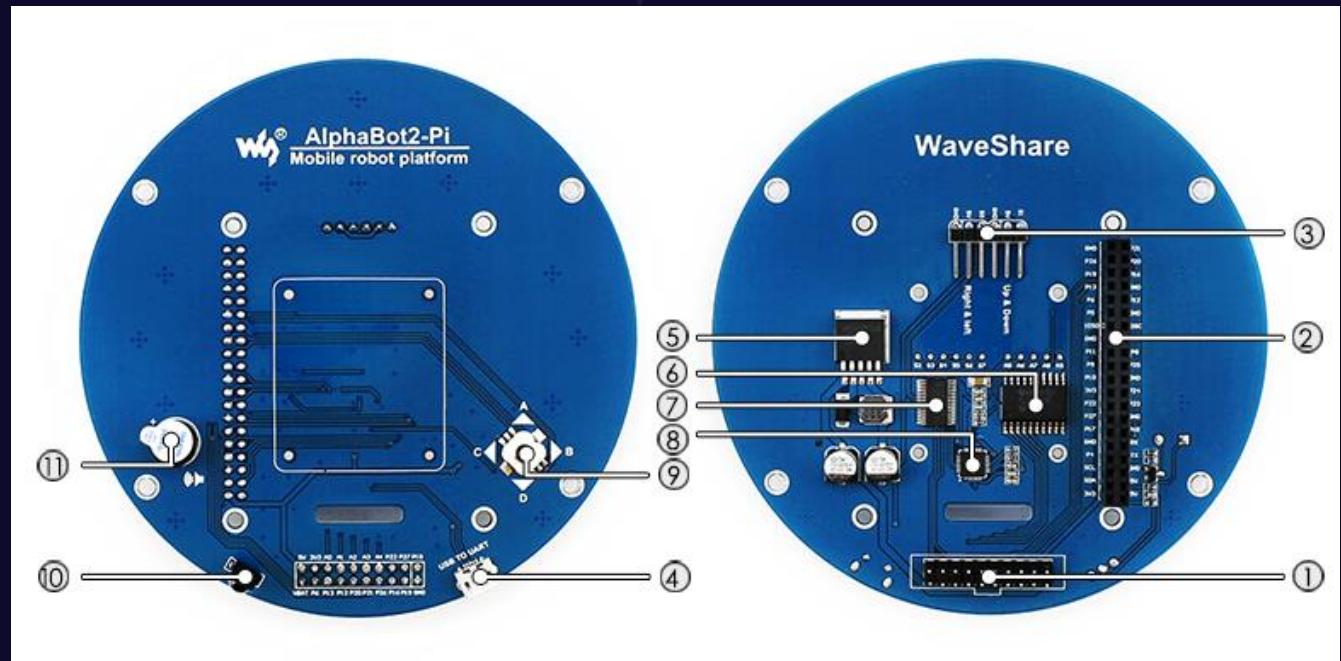
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Apr  1 06:44:26 2023 from 192.168.1.16
pi@E1163153:~ $ |
```



A closer look at the hardware

6. TLC1543: 10-bit ADC acquisition chip

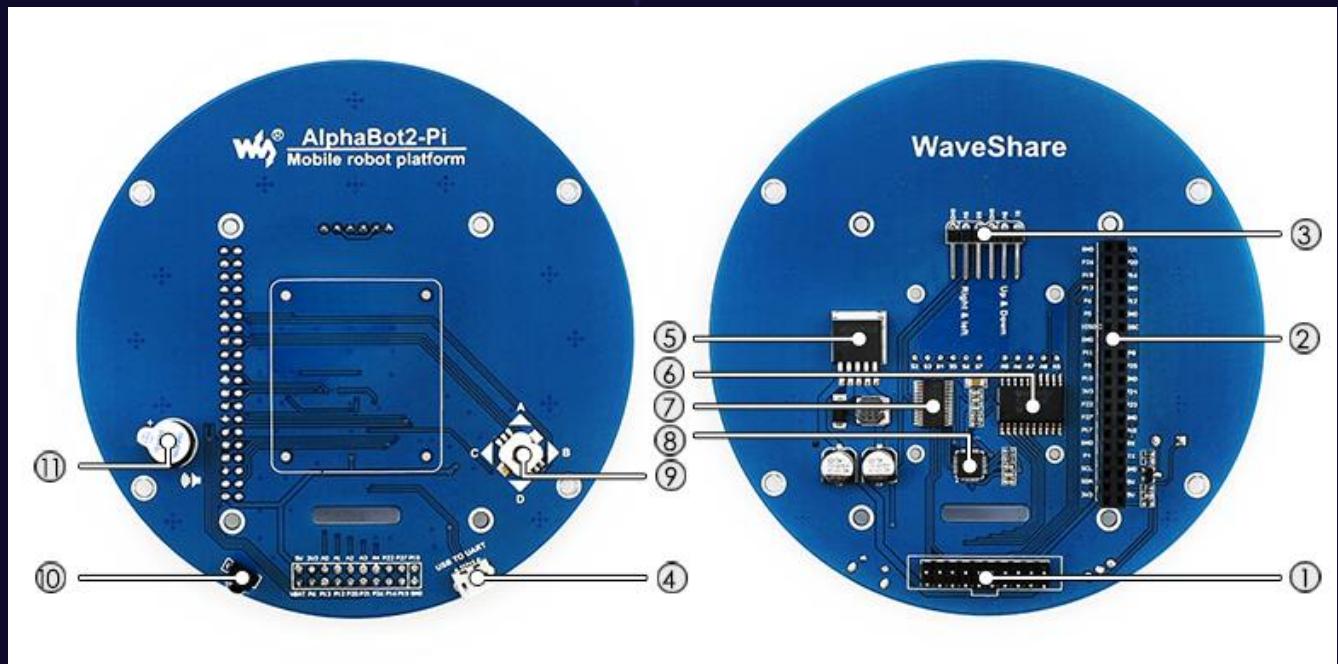
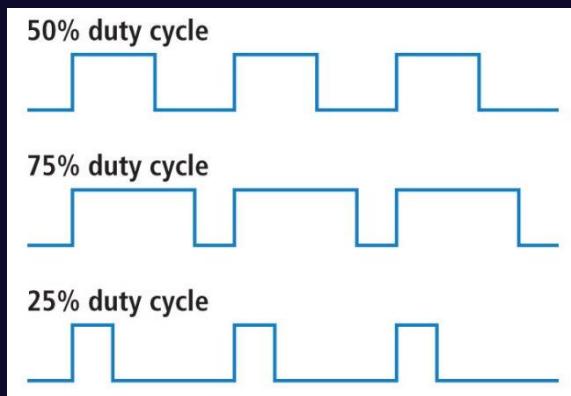
- ADC (Analog to Digital Controller)
- Allows the Pi to use analog sensors
- Voltage signal from a sensor might range from 0 to 5 volts
- Digital (0 or 1)



A closer look at the hardware

7 - PCA9685: servo controller

- Allows Pan and Tilt Camera to move
- 16-Channel 12-Bit PWM (Pulse Width Modulation) Servo Driver
- Power delivered to motor by switching it on and off rapidly.

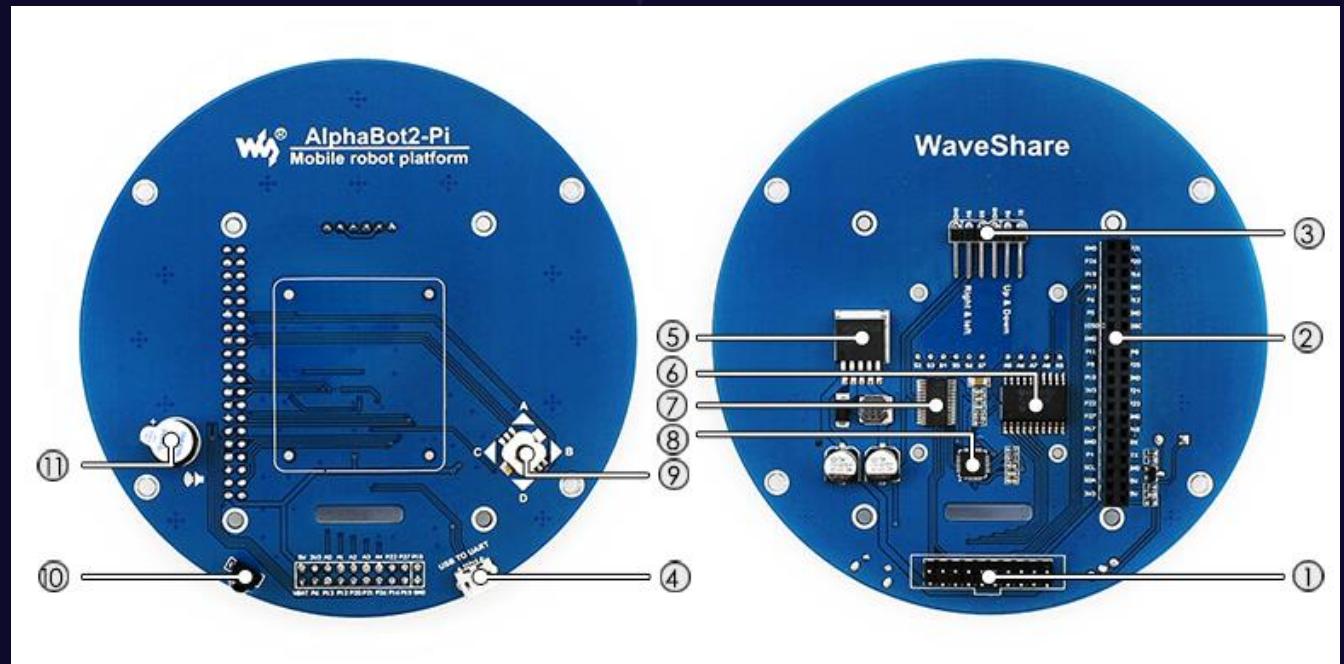


A closer look at the hardware

9. Joystick

- 4 Way directional control
- Center button

Allows you to go: Pew Pew! 😊

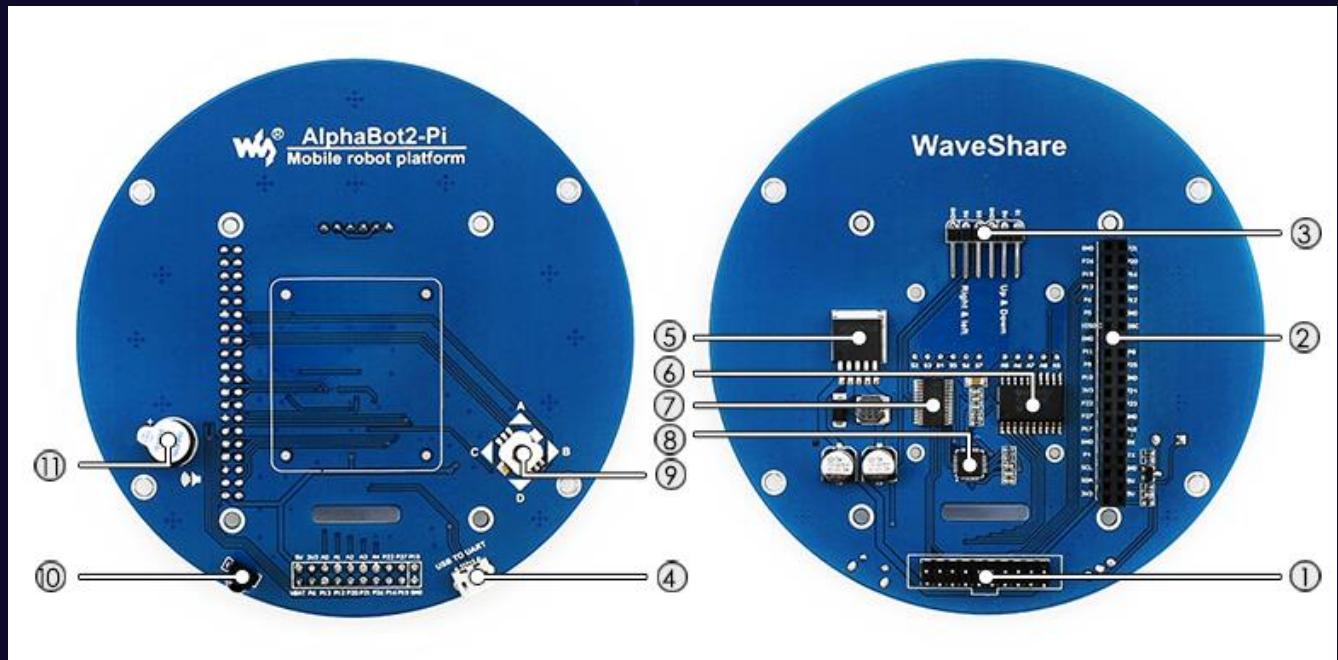


```
pi@E1163153:~ $ cd IoTHackDaysSession2/AlphabotTest/  
pi@E1163153:~/IoTHackDaysSession2/AlphabotTest $ python Joystick.py
```

A closer look at the hardware

10 IR receiver

- IR (Infrared) receiver
- Receives and decode signals sent from an IR remote control.
- Photodiode
 - Converts infrared light into an electrical signal
- Preamplifier
 - Amplifies the signal
- Demodulator
 - Removes noise or interference
- Decoder
 - Translates the signal into a digital code

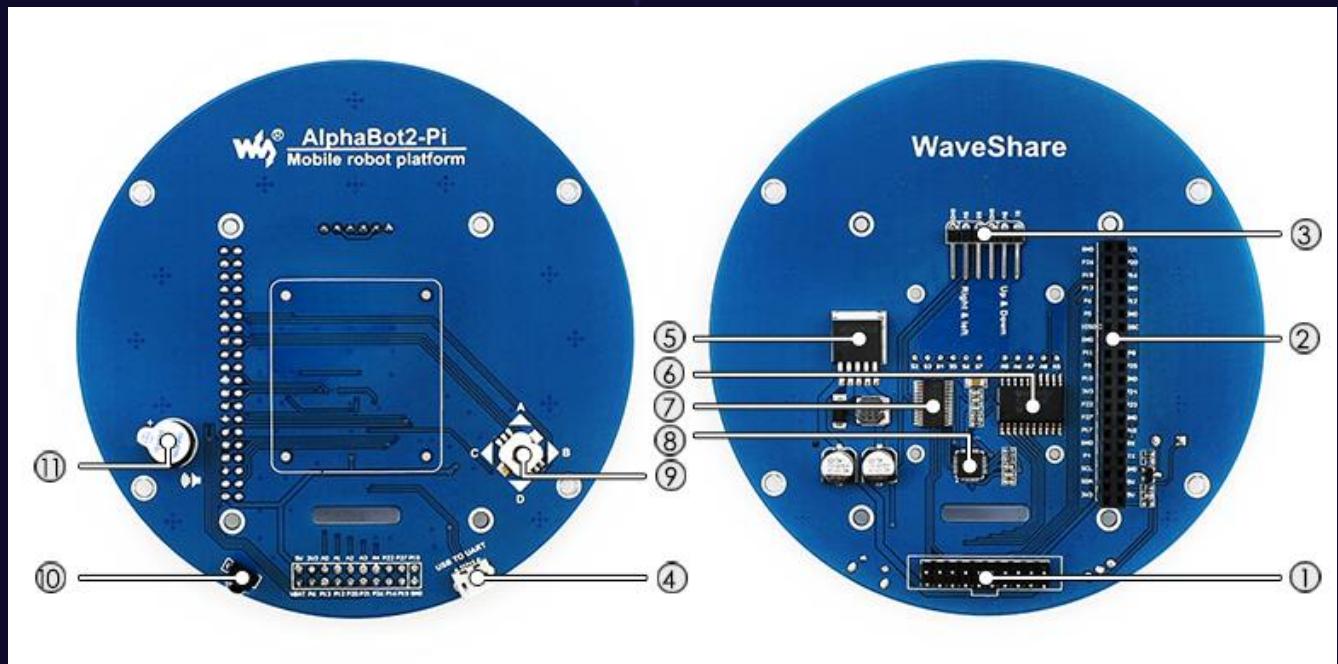


```
pi@E1163153:~/IoTHackDaysSession2/AlphabotTest $ python IRremote.py
IRremote Test Start ...
```

A closer look at the hardware

11. Buzzer

- Allows you to raise audible alarms or give feedback

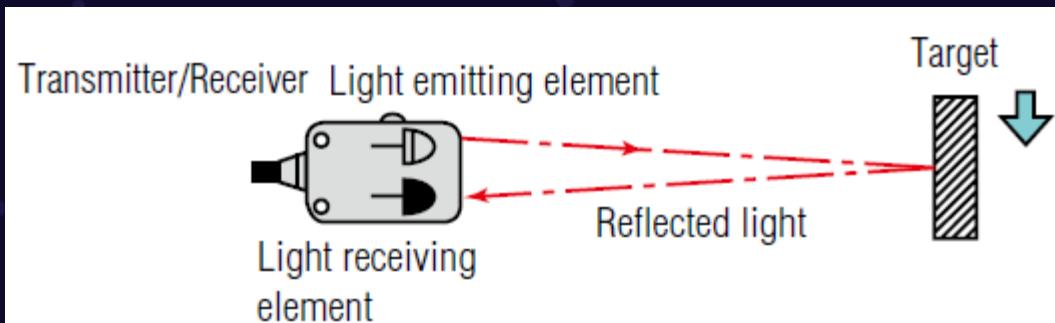
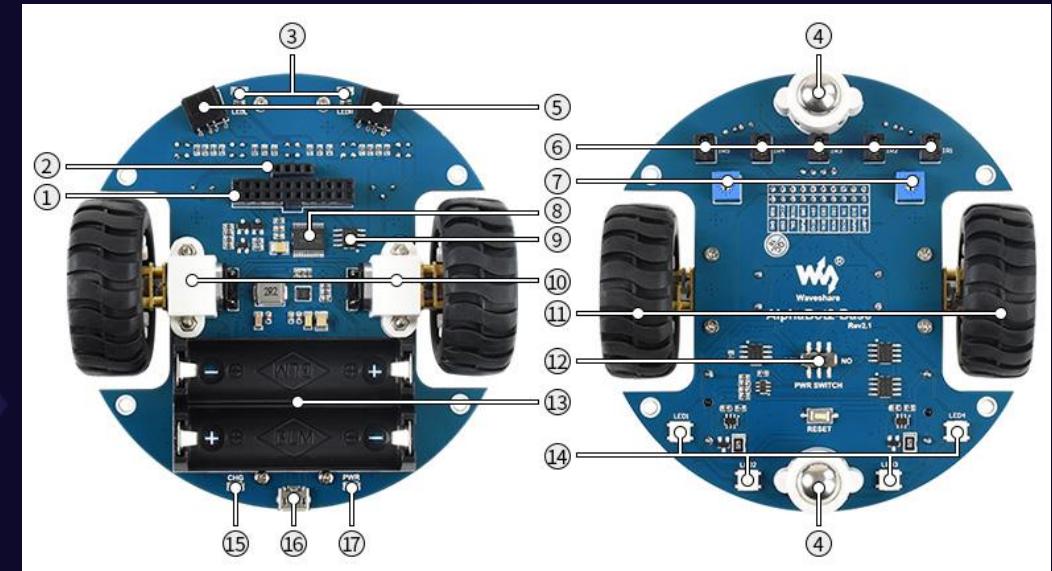


A closer look at the hardware

3 Obstacle avoiding indicators

5 ST188: Reflective infrared photoelectric sensor

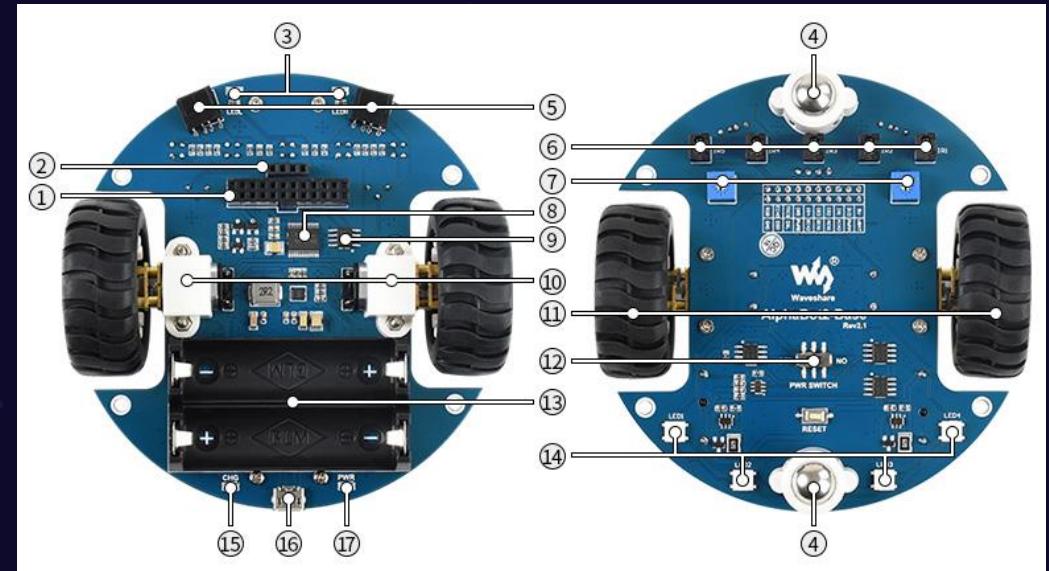
- Obstacle avoiding indicator – LED
- Reflective infrared photoelectric sensor bounces light of an object
 - Infrared emitter (LED)
 - Receiver (photodiode)



A closer look at the hardware

6. ITR20001/T: reflective infrared photoelectric sensor

- Detect the presence or absence of an object.
- Works by emitting a beam of light and then measuring the amount of light that is reflected back
- Used for line tracking
 - Detects light and dark

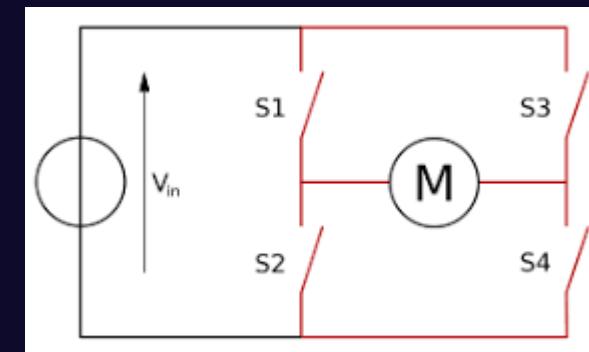
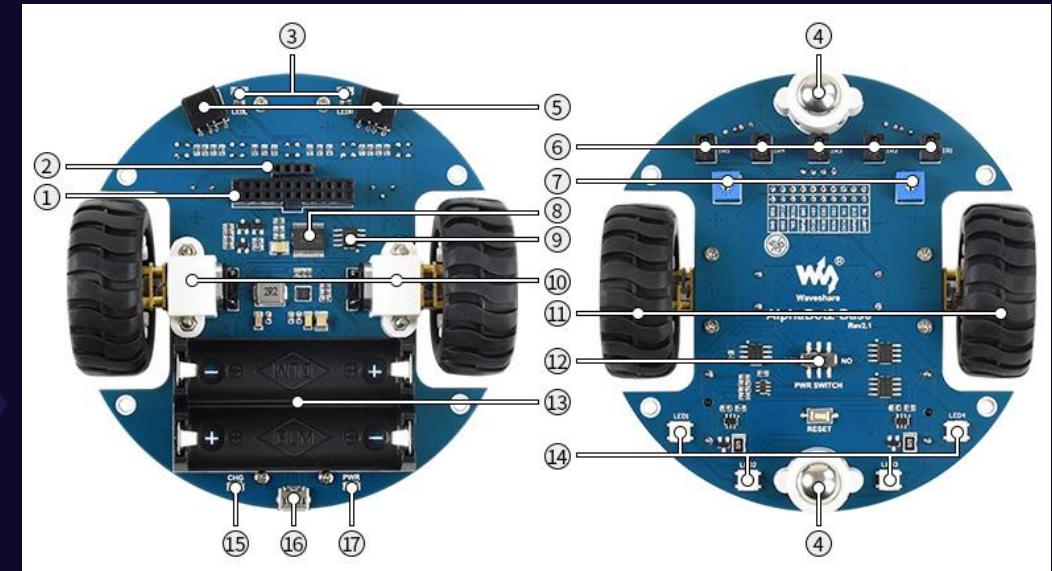


```
pi@E1163153:~/IoTHackDaysSession2/AlphabotTest $ python TRSensors.py
TRSensors Example
[776, 926, 936, 949, 795]
[777, 926, 937, 950, 795]
```

A closer look at the hardware

8. TB6612FNG dual H-bridge motor driver

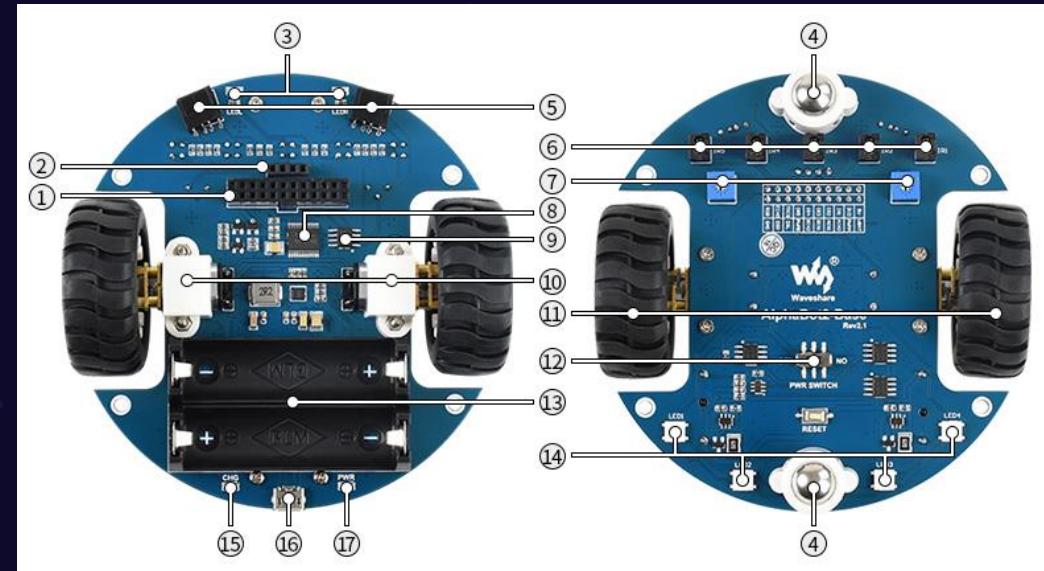
- Controls the direction and speed of a DC motor
- Four switches arranged in the shape of an "H"
- Two pairs, each pair controlling the current flow to the motor in either direction
- One pair are turned on and other pair are turned off, current flows in one direction through the motor.
- First pair are turned off and second pair are turned on, current flows in the opposite direction through the motor.



A closer look at the hardware

14. WS2812B: true-color RGB LEDs

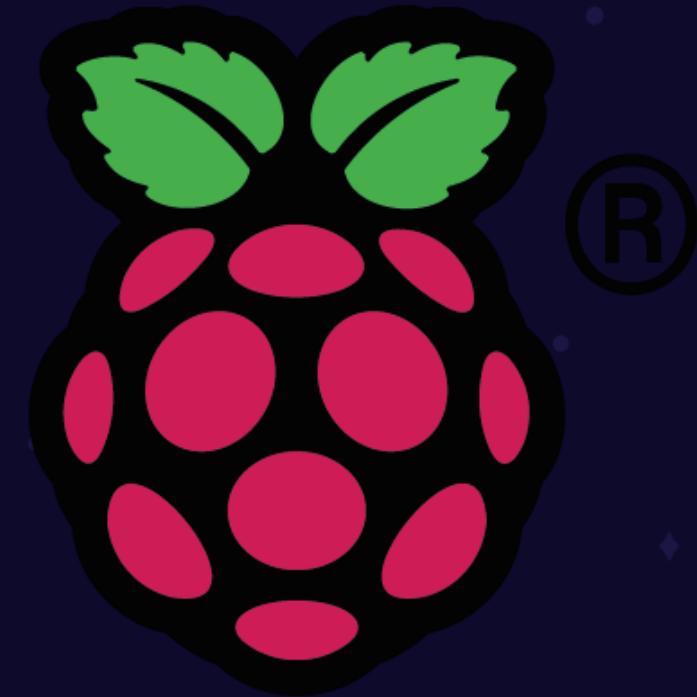
- Addressable RGB LED Strip
- Multiple light-emitting diodes (LEDs)
- RGB (Red, Green, Blue)
- (PWM) function that can control the brightness



```
pi@E1163153:~/IoTHackDaysSession2/AlphabotTest $ sudo python LightShow.py
Press Ctrl-C to quit.
Use "-c" argument to clear LEDs on exit
Color wipe animations.
Theater chase animations.
```

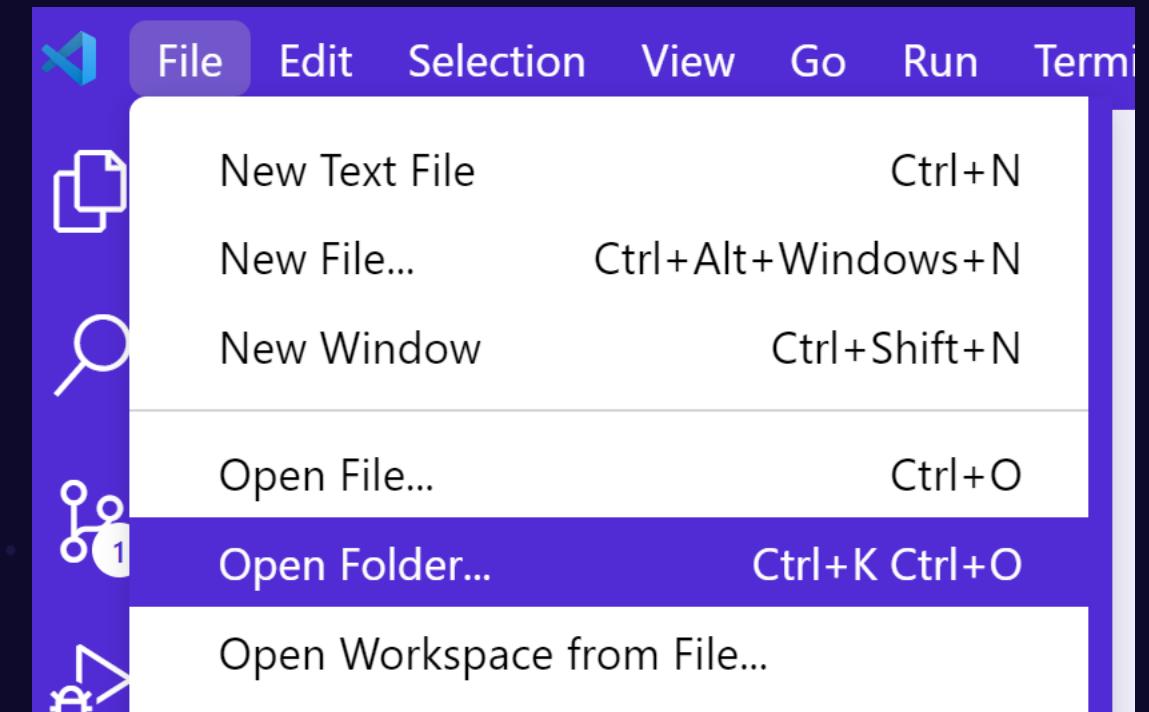
Module 7

The Alphabot 2 - .NET



Try it out!

In VS Code open the folder
Alphabot2Dotnet from the code
cloned from Github



Git clone <https://github.com/CPTMSDUG/IoTHackDaysSession2DotNet>

Controlling the Motors (PWM)

```
private GpioController _controller;

    _pwmA = new SoftwarePwmChannel(_ena, frequency: 500, usePrecisionTimer: true, dutyCycle: _pa,
controller: _controller);
    _pwmB = new SoftwarePwmChannel(_enb, frequency: 500, usePrecisionTimer: true, dutyCycle: _pb,
controller: _controller);
    _pwmA.Start();
    _pwmB.Start();

public void Stop()
{
    _pwmA.DutyCycle = 0;
    _pwmB.DutyCycle = 0;

    _controller.Write(_ain1, PinValue.Low);
    _controller.Write(_ain2, PinValue.Low);
    _controller.Write(_bin1, PinValue.Low);
    _controller.Write(_bin2, PinValue.Low);
}
```

Controlling the Motors (PWM)

```
private GpioController _controller;

    _pwmA = new SoftwarePwmChannel(_ena, frequency: 500, usePrecisionTimer: true, dutyCycle: _pa,
controller: _controller);
    _pwmB = new SoftwarePwmChannel(_enb, frequency: 500, usePrecisionTimer: true, dutyCycle: _pb,
controller: _controller);
    _pwmA.Start();
    _pwmB.Start();

public void Forward()
{
    _pwmA.DutyCycle = _pa;
    _pwmB.DutyCycle = _pb;
    _controller.Write(_ain1, PinValue.Low);
    _controller.Write(_ain2, PinValue.High);
    _controller.Write(_bin1, PinValue.Low);
    _controller.Write(_bin2, PinValue.High);

    Console.WriteLine("Forward");
}
```

Controlling the Motors (PWM)

```
public void Left()
{
    _pwmA.DutyCycle = 0.5;
    _pwmB.DutyCycle = 0.5;
    _controller.WriteLine(_ain1, PinValue.High);
    _controller.WriteLine(_ain2, PinValue.Low);
    _controller.WriteLine(_bin1, PinValue.Low);
    _controller.WriteLine(_bin2, PinValue.High);

    Console.WriteLine("Left");
}

public void Right()
{
    _pwmA.DutyCycle = 0.5;
    _pwmB.DutyCycle = 0.5;
    _controller.WriteLine(_ain1, PinValue.Low);
    _controller.WriteLine(_ain2, PinValue.High);
    _controller.WriteLine(_bin1, PinValue.High);
    _controller.WriteLine(_bin2, PinValue.Low);

    Console.WriteLine("Right");
}
```

Git clone <https://github.com/CPTMSDUG/IoTHackDaysSession2DotNet>

Controlling the Buzzer

```
const int BuzzerPin = 4;

_gpioController.OpenPin(BuzzerPin, PinMode.Output);

public void Buzz()
{
    _gpioController.Write(BuzzerPin, PinValue.High);
}

public void Silence()
{
    _gpioController.Write(BuzzerPin, PinValue.Low);
}
```

Controlling the Led Strips (WS2812)

```
var controller = settings.AddController(16, Pin.Gpio18, StripType.WS2812_STRIP, ControllerType.PWM0, 255, false);

using (var rpi = new WS281x(settings))
{
    var colors = GetAnimationColors();
    while (cycleIndex < colorCycles)
    {

        for (int i = 0; i <= controller.LEDCount - 1; i++)
        {
            var colorIndex = (i + colorOffset) % colors.Count;
            controller.SetLED(i, colors[colorIndex]);
        }

        rpi.Render();

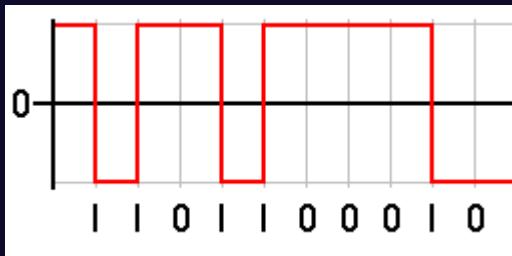
        if (colorOffset == int.MaxValue)
        {
            colorOffset = 0;
        }
        colorOffset++;
        Thread.Sleep(50);

        cycleIndex++;
    }
}
```

Git clone <https://github.com/CPTMSDUG/IoTHackDaysSession2DotNet>

Controlling the Joystick

.NET Events and Event Handlers on Rising or Falling Pins



```
_gpioController = controller;

_gpioController.OpenPin(Ctr, PinMode.InputPullUp);
_gpioController.OpenPin(A, PinMode.InputPullUp);
_gpioController.OpenPin(B, PinMode.InputPullUp);
_gpioController.OpenPin(C, PinMode.InputPullUp);
_gpioController.OpenPin(D, PinMode.InputPullUp);

_gpioController.RegisterCallbackForPinValueChangedEvent(Ctr,
PinEventTypes.Falling, CenterPressed);
_gpioController.RegisterCallbackForPinValueChangedEvent(A,
PinEventTypes.Falling, APressed);
_gpioController.RegisterCallbackForPinValueChangedEvent(B,
PinEventTypes.Falling, BPressed);
_gpioController.RegisterCallbackForPinValueChangedEvent(C,
PinEventTypes.Falling, CPressed);
_gpioController.RegisterCallbackForPinValueChangedEvent(D,
PinEventTypes.Falling, DPressed);
}

private void APressed(object sender, PinValueEventArgs
pinValueEventArgs)
{
    _alphabot.Forward();
}

private void BPressed(object sender, PinValueEventArgs
pinValueEventArgs)
{
    _alphabot.Right();
}
```

Controlling the Collision Detector

```
_gpioController.OpenPin(DR, PinMode.InputPullUp);
_gpioController.OpenPin(DL, PinMode.InputPullUp);

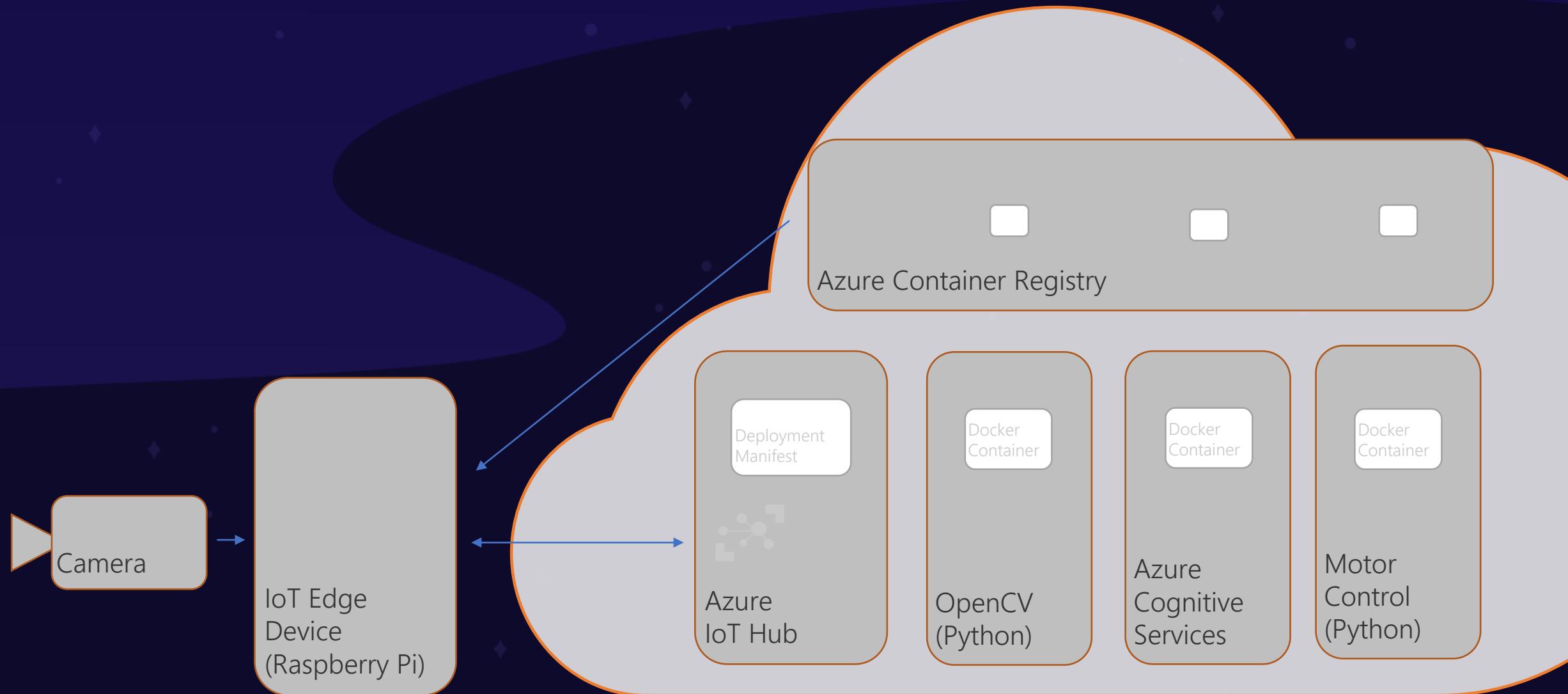
_gpioController.RegisterCallbackForPinValueChangedEvent(DL, PinEventTypes.Falling, DetectLeftCallback);
_gpioController.RegisterCallbackForPinValueChangedEvent(DR, PinEventTypes.Falling, DetectRightCallback);

}

private void DetectLeftCallback(object sender, PinValueChangedEventArgs pinValueChangedEventArgs)
{
    LeftCollision?.Invoke(this, pinValueChangedEventArgs);
    Collision?.Invoke(this, pinValueChangedEventArgs);
}

private void DetectRightCallback(object sender, PinValueChangedEventArgs pinValueChangedEventArgs)
{
    RightCollision?.Invoke(this, pinValueChangedEventArgs);
    Collision?.Invoke(this, pinValueChangedEventArgs);
}
```

Next Session – Azure IoT Edge and AI

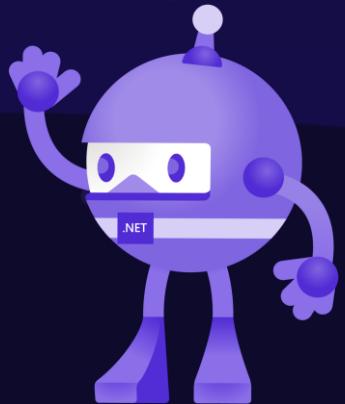




Resources

- Twitter: [@adpead](https://twitter.com/adpead)
- About.me: https://about.me/allan_pead
- Linkedin: <https://www.linkedin.com/in/adpead/>
- Blog: <https://explorationspace.co.za>
- Raspberry Pi South Africa
- <https://www.facebook.com/groups/1493503984198019>

Thank you!!



Getting Started with .NET

<https://learn.microsoft.com/en-us/dotnet/standard/get-started>

Visual Studio Remote Debugger

<https://www.hanselman.com/blog/remote-debugging-with-vs-code-on-windows-to-a-raspberry-pi-using-net-core-on-arm>

Getting Started with Alphabot

<https://www.waveshare.com/wiki/AlphaBot2>

Azure IoT Central

<https://azure.microsoft.com/en-us/products/iot-central/>

