

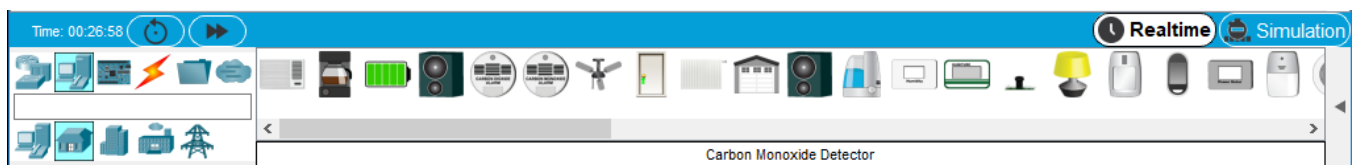
## P11: Networking with IoT

### Q1: Understand the Basics of IoT

Where are the Things?

In addition to routers and switches, the Network Component Box now contains a variety of Smart Things and components. They can be created by selecting the device and then clicking on a blank area either on the Logical Workspace or the Physical Workspace.

**Smart Things** are physical objects that can connect to the Registration Server or Home Gateway through a network interface. They are separated into subcategories under End Devices. The categories consist of Home, Smart City, Industrial, and Power Grid. Pictured below are a few Smart Thing devices that are in the Home category.

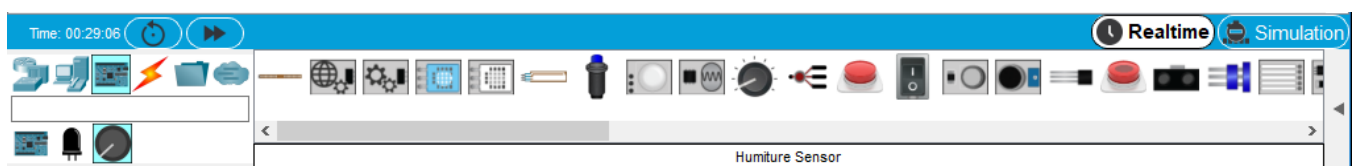


**Components** are physical objects that connect to microcontroller (MCU-PT) or single boarded computers (SBC-PT) and typically does not have a network interface. These are simple devices that only communicate through their analog or digital slots.

There are three subcategories for Components, they are:

- **Boards:** microcontrollers (MCU-PT), single boarded computers (SBC-PT), and a special device called Thing. Things are used to create self-contained physical objects like coffee makers or smoke alarms.
- **Actuators:** these components manipulate the Environment, themselves, or the area around them.
- **Sensors:** these components sense the Environment (photo detectors, temperature sensor), the area around them (RFID, metal sensor), or interactions (potentiometer, push button).

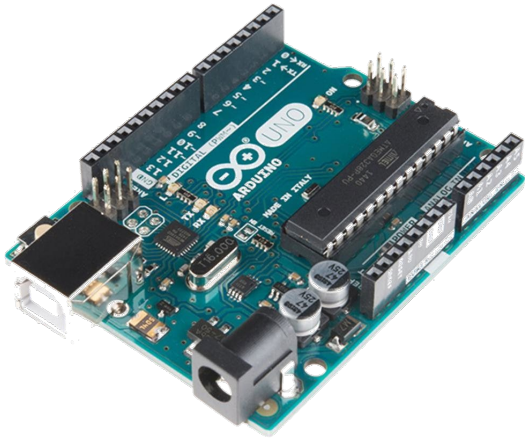
Below is an image some sensor components:



## Examples for MCU (Microcontroller Unit) in real world

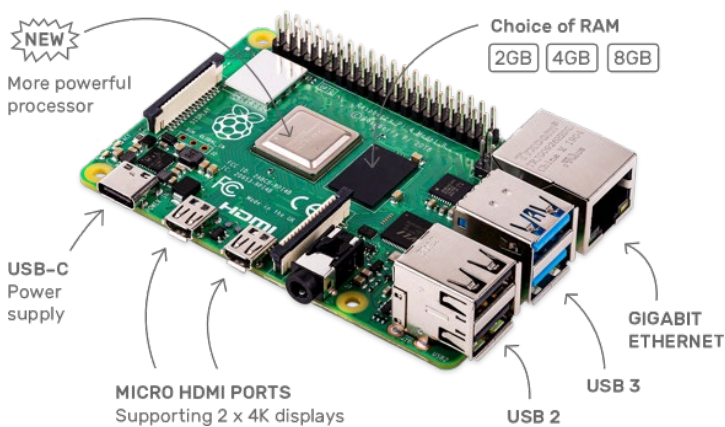
### Arduino

(Note: There is **NO CPU** – central processing unit but an IC (Integrated Circuit) component on the board. For heavy CPU or GPU intensive work, use a SBC.)



## Examples for SBC (Single Board Computers) in real world

### Raspberry Pi 4



### Nvidia Jetson (CPU + GPU for AI (Artificial Intelligence) tasks)



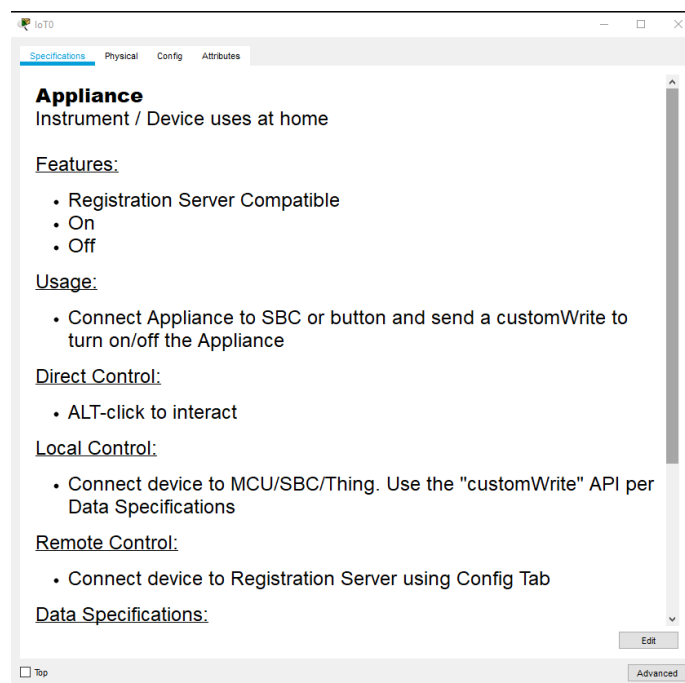
## Interacting with Smart Things

There are many ways to interact with Things. The Specifications tab of the Device Dialog will list the types of interactions that are allowed for that specific Thing.

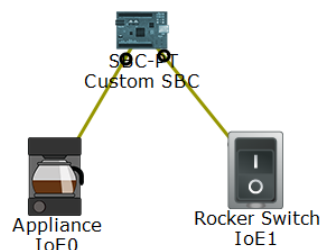
The types of interactions are:

- **Direct Control:** interact with the object directly on the workspace using the mouse. The mouse interactions are usually modified by pressing "ALT" on the keyboard at the same time as moving the mouse or clicking.
- **Local Control:** interact with the object using the digital or analog slots.
- **Remote Control:** interact with the object over an IP network. In most cases, this means interacting with the Registration Server or Home Gateway.

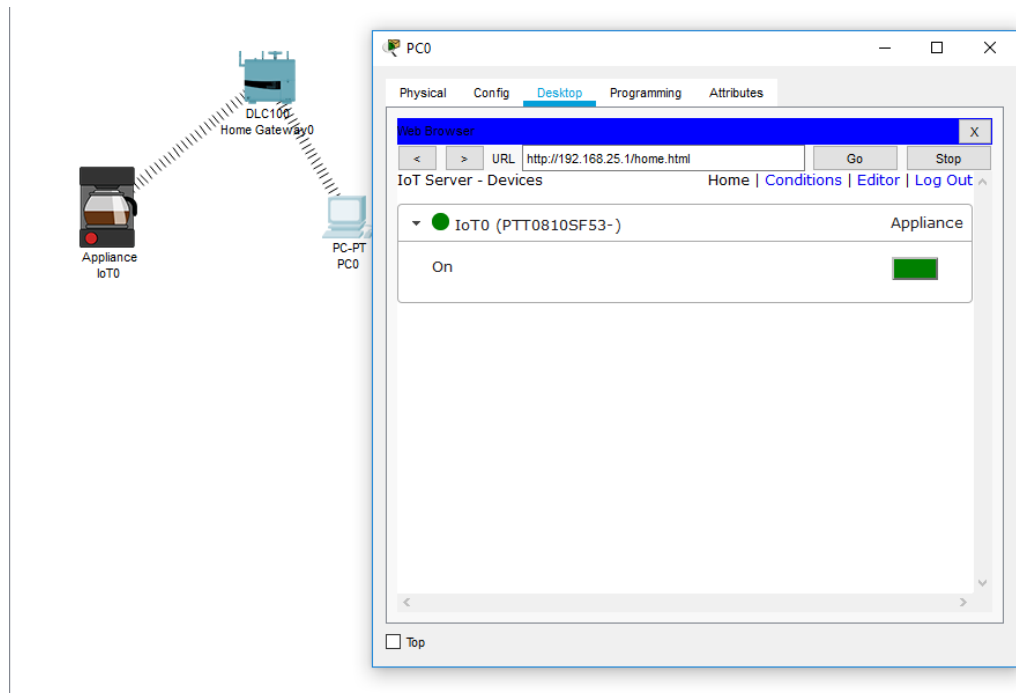
Below is a sample specification for the prepackaged Thing, Appliance. In this case, ALT-clicking directly on the Appliance in the workspace will turn it on.



It is also possible to control the Appliance locally through the digital and analog ports as shown in the image below. The SBC sends an on or off command to the Appliance depending on the Rocker Switch state.

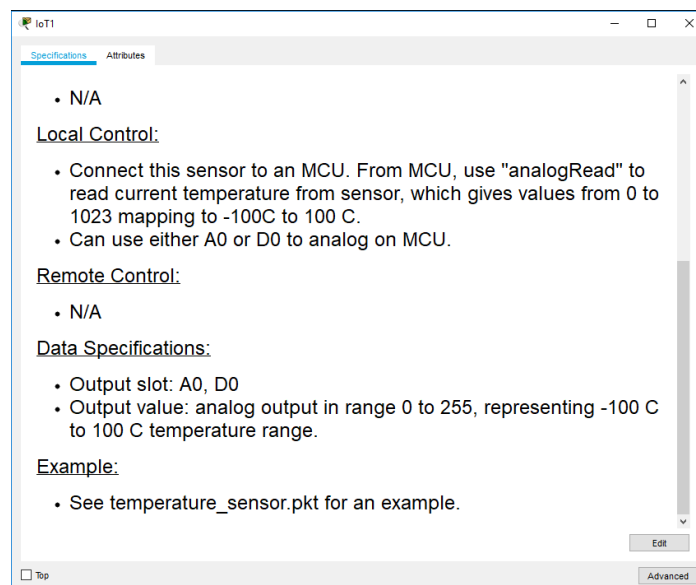


Finally, it is possible to remotely control the Thing through the Registration Server or Home Gateway. The Appliance is registered to the DLC100 Home Gateway. The Home Gateway also provides wireless connectivity to the PC and Appliance. Using the PC and connecting to the Home Gateway web service, it is possible to remotely control the Appliance over the network.



### Interacting with Components

Components do not have network interface cards. Typically, they are connected to the digital or analog slots on the MCU or SBC. It is important to read the Specifications page for the component because it typically describes the data specifications. For example, see the specification for the Temperature Sensor below:



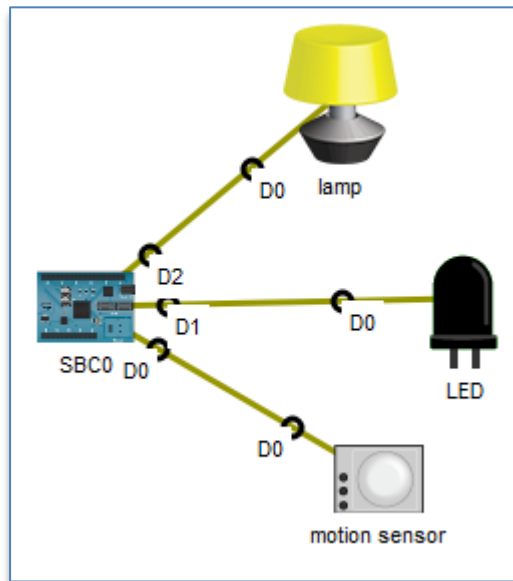
## Cabling Things

Cabling Things is similar to cabling networking devices. Please refer to the Workspace Basics, Logical Workspace for more information. The main difference is the addition of digital and analog slots (D0, D1, A0, A1, etc). These slots only accept the IoE Custom Cable. The IoE Custom Cable is a bundle of wires, which are inaccessible to the user, that provides all the ground, power, and data connections necessary for communication between all Things, components, MCU, and SBCs.

Currently, the model does not let you unwrap the IoE Custom cable and create circuit schematics.

## Q2: Configuring a simple network of things

- On a new **.pkt** file, implement the simple IoT network shown below:



Please connect the devices to the correct pins (D0, D1, D2) of SBC0 as shown on the diagram above.

### a. Devices

Device Type	Device Name
SBC Board	SBC0
LED	LED
Light	lamp
Motion Sensor	motion sensor

Use PT Official Help to find the location of the IoT devices on Packet Tracer device bar.

To read the specification of each device (API to use, device information, states, etc.) click on the Device.

- Go to SBC->Programming
- Create a project named "Motion-Python"
- Write the code below:

```
from gpio import *
from time import *

motion = 0

# Read from a the motion sensor
def readFromSensor():
    global motion
    motion = digitalRead(0) # Read from slot 0 using digitalRead API
    print "readFromSensor(): motion " + str(motion)

def writeToDevices():
    global motion
    print "writeToDevices(): motion " + str(motion)

    if motion == 1023:
        print "HIGH"
        # Write the Lamp state "2" (2=ON, 1=Dim, 0=Off) on slot 2 using customWrite API. Click on the lamp to see the details of the states
        customWrite(2, 2)

        # Write the LED state "1023" or HIGH (1023=HIGH, 0=Off) on slot 1 using analogWrite API
        analogWrite(1, HIGH)
    else:
        print "LOW"
        customWrite(2, 0)
        analogWrite(1, LOW)

def main():
    # Initialize the connected pins of SBC
    pinMode(0, IN)
    pinMode(1, OUT)
    pinMode(2, OUT)

    while True:
        readFromSensor()
        writeToDevices()
        delay(5000) # wait for some time (i.e. 5 sec) before reading motion sensor input

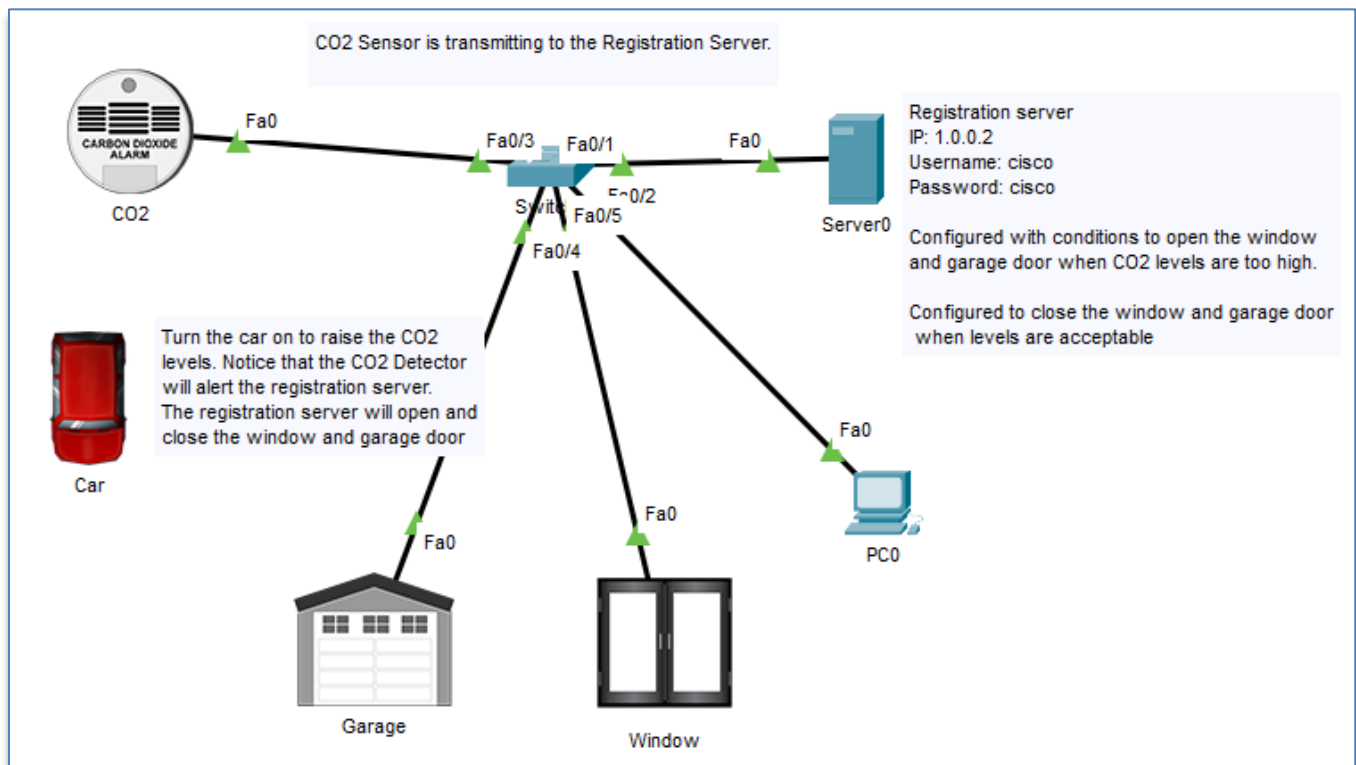
if __name__ == "__main__":
    main()
```

- e. **Run** the project
- f. Press **Alt** key and move the mouse on top of the motion sensor, wait for 5 seconds
  - i. Lamp and LED must be turned on
  - ii. Wait another 5 seconds without moving the mouse over the sensor
  - iii. Lamp and LED must be turned off.
  - iv. Observe the console prints of the program to better understand code
- g. Switch to simulation mode and observe signal transmission.



### Q3: Configuring a IoE (Internet of Everything) Registration Server

- On a new .pkt file, implement the IoT network shown below:



- a. In order to communicate with a **Registration Server**, the devices have to be configured with an IP address. Use the following table to assign the IP addresses accordingly

Device Type	Device Name	IP	Gateway (optional)
PT-Server	Server0	1.0.0.2/255.0.0.0	1.0.0.1
Carbon Dioxide Detector	CO2	1.0.0.3/255.0.0.0	1.0.0.1
Garage Door	Garage	1.0.0.4/255.0.0.0	1.0.0.1
Window	Window	1.0.0.5/255.0.0.0	1.0.0.1
PT-PC	PC0	1.0.0.5/255.0.0.0	1.0.0.1

- b. Go to **Server0** (Registration Server) and enable **IoT Service** (Services -> IoT Service)
- c. Open the web browser on **PC0** and type 10.0.0.2 to access the **Server0** (IoT Registration Server) to create a new account on **Server0**
- username:** cisco, **password:** cisco
  - Once the account is created, Go to **Server0** -> Services -> IoT, see whether the newly created account is shown there with the credentials.
- d. Devices must be configured to communicate with the **Server0** (IoT Registration Server) on the network as below:

Device Type	Device Name	Settings -> IOT Server
Carbon Dioxide Detector	CO2	Enable <b>Remote Server</b> Server Address: 1.0.0.2

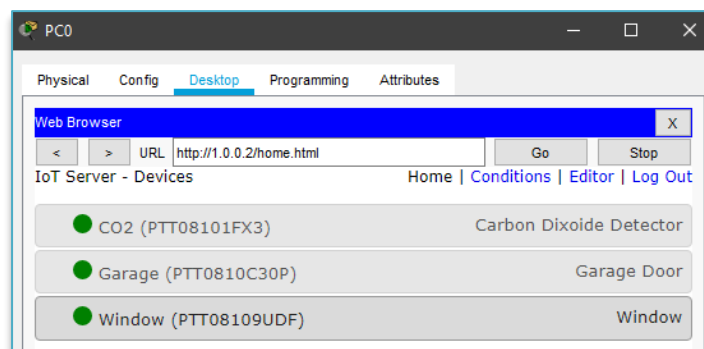
		Username: cisco Password: cisco
Garage Door	Garage	Enable <b>Remote Server</b> Server Address: 1.0.0.2 Username: cisco Password: cisco
Window	Window	Enable <b>Remote Server</b> Server Address: 1.0.0.2 Username: cisco Password: cisco

e. Lets configure the logical conditions to open the garage door and window once a hazardous level of CO2 is detected.

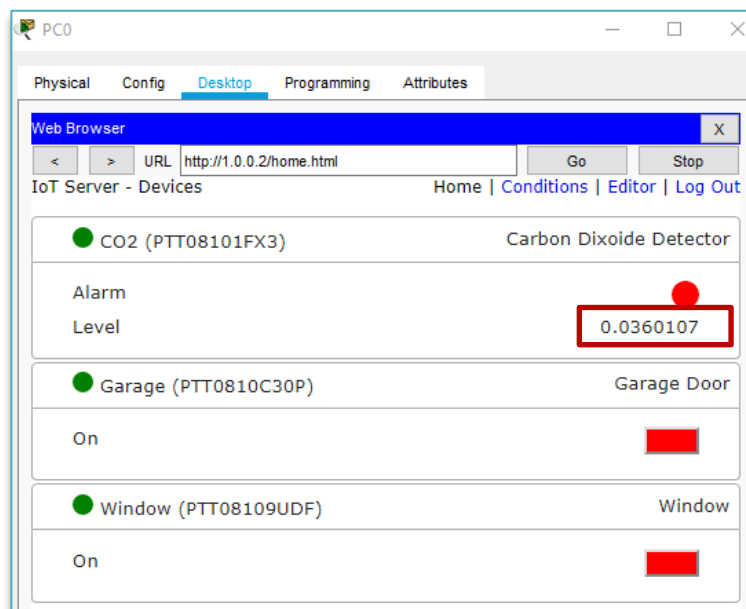
i. Access the web portal of the server from **PC0** (via the web browser)

Click on the link Home

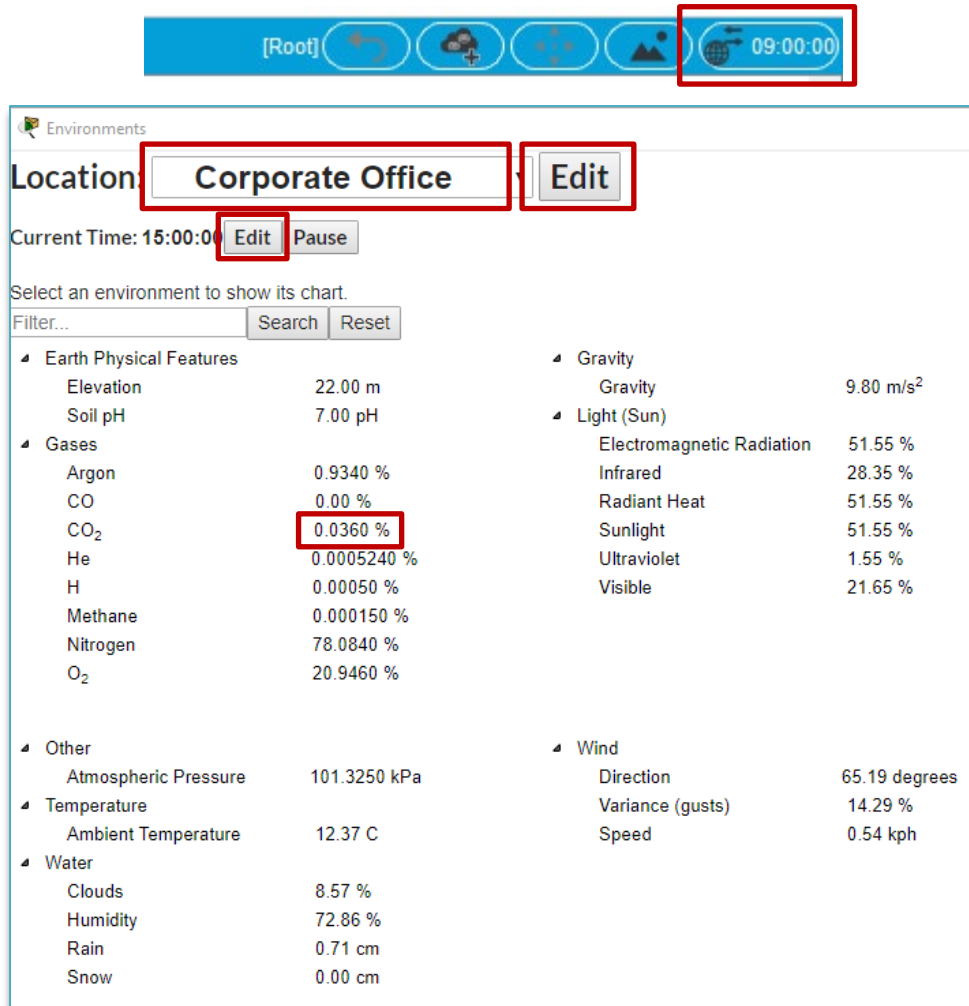
The IoT devices must be shown as below: (Click on the device name below to show the status)



Click on the device name to show the status (current CO2 reading, etc.)



Validate the CO2 level of the container where all these devices are placed. Click on **Environments** button:  
(Physical Layer Container: Cooperate Office)



Environments

Location: **Corporate Office** [Edit](#)

Current Time: 15:00:00 [Edit](#) [Pause](#)

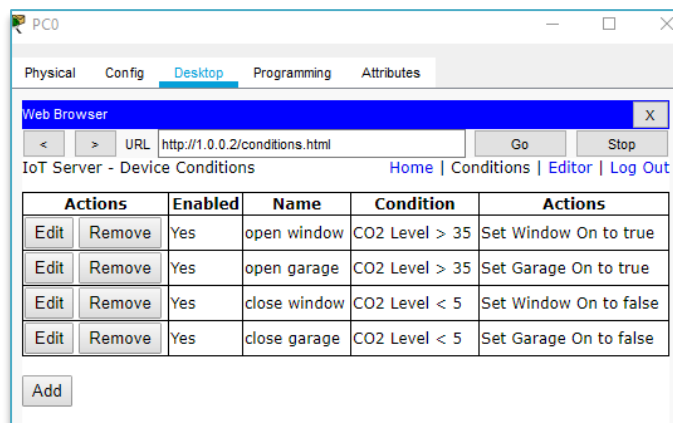
Select an environment to show its chart.

Filter... [Search](#) [Reset](#)

- Earth Physical Features
  - Elevation: 22.00 m
  - Soil pH: 7.00 pH
- Gases
  - Argon: 0.9340 %
  - CO: 0.00 %
  - CO<sub>2</sub>: **0.0360 %**
  - He: 0.0005240 %
  - H: 0.00050 %
  - Methane: 0.000150 %
  - Nitrogen: 78.0840 %
  - O<sub>2</sub>: 20.9460 %
- Other
  - Atmospheric Pressure: 101.3250 kPa
- Temperature
  - Ambient Temperature: 12.37 C
- Water
  - Clouds: 8.57 %
  - Humidity: 72.86 %
  - Rain: 0.71 cm
  - Snow: 0.00 cm
- Gravity
  - Gravity: 9.80 m/s<sup>2</sup>
- Light (Sun)
  - Electromagnetic Radiation: 51.55 %
  - Infrared: 28.35 %
  - Radiant Heat: 51.55 %
  - Sunlight: 51.55 %
  - Ultraviolet: 1.55 %
  - Visible: 21.65 %
- Wind
  - Direction: 65.19 degrees
  - Variance (gusts): 14.29 %
  - Speed: 0.54 kph

*Note: You can click on Edit (next to location) to simulate different environment conditions with key points. In PT there IoT Devices that will detect these environmental changes (CO<sub>2</sub>, CO, H, O<sub>2</sub>, Visible Light, etc.).*

- ii. Click on the link [Conditions](#) and add following conditions



PC0

Physical Config **Desktop** Programming Attributes

Web Browser [X](#)

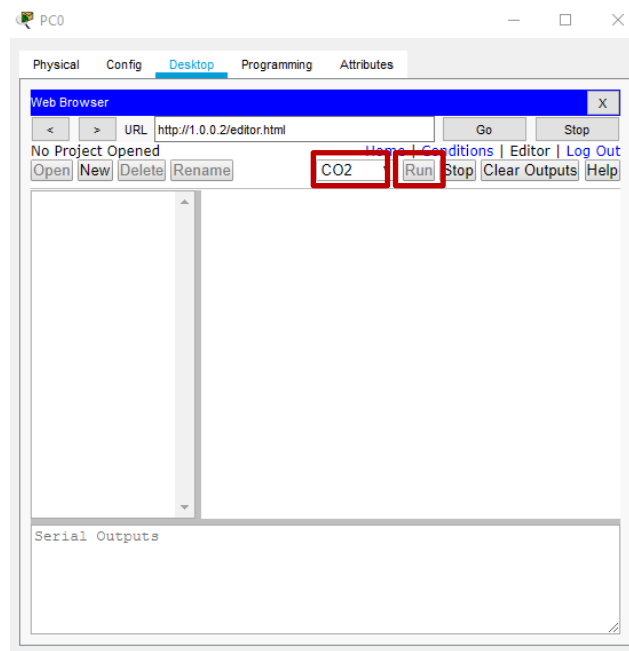
< > URL <http://1.0.0.2/conditions.html> [Go](#) [Stop](#)

IoT Server - Device Conditions [Home](#) | [Conditions](#) | [Editor](#) | [Log Out](#)

Actions		Enabled	Name	Condition	Actions
<a href="#">Edit</a>	<a href="#">Remove</a>	Yes	open window	CO <sub>2</sub> Level > 35	Set Window On to true
<a href="#">Edit</a>	<a href="#">Remove</a>	Yes	open garage	CO <sub>2</sub> Level > 35	Set Garage On to true
<a href="#">Edit</a>	<a href="#">Remove</a>	Yes	close window	CO <sub>2</sub> Level < 5	Set Window On to false
<a href="#">Edit</a>	<a href="#">Remove</a>	Yes	close garage	CO <sub>2</sub> Level < 5	Set Garage On to false

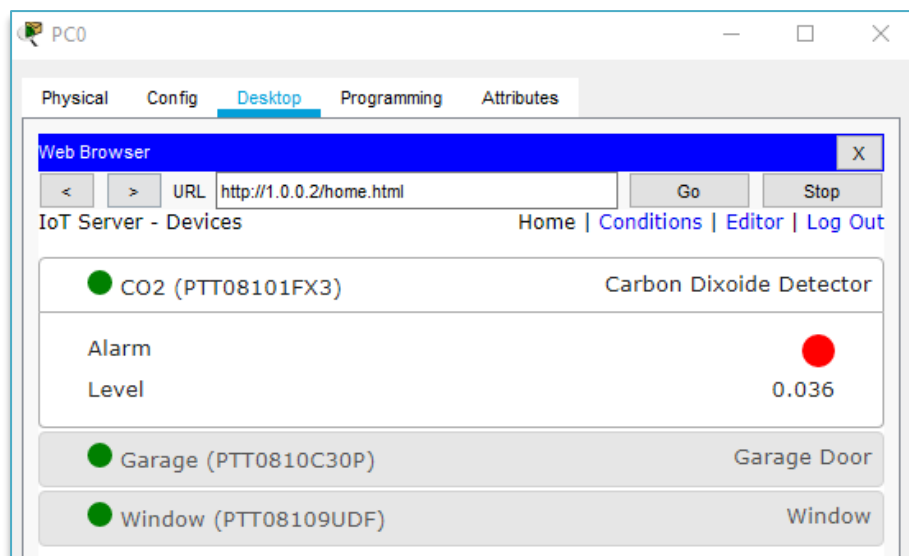
[Add](#)

- iii. Optional – Click on the link [Editor](#). This will enable you to write programs on individual devices to work with custom operations defined on them. For this activity, we skip this part.



*Note: Make sure the devices are in Run state (Active state)*

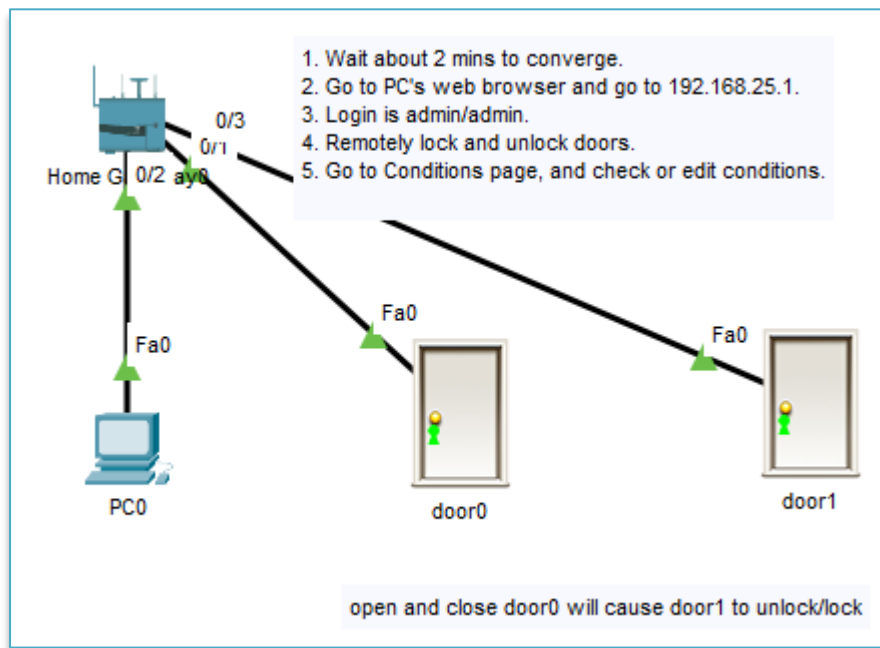
- iv. Go back to Home and expand the CO2 Sensor detail tab, keep it open where visible before you do the next step. Watch closely on CO2 Level. (You can also view the CO2 level of current physical container by going to Environments windows shown above)



- f. Press Alt and click on the Car. Observe what happens next! 😊

#### Q4: Configuring a Home Gateway

- On a new **.pkt** file, implement the IoT network shown below:



*Note: Home Gateway is like a Router that connect the local network to the internet (Ports: 4 LAN ports, 1 internet port). But it has IoT Registration Server in built (go to GUI tab and enable HTTP) and can be accessed via a web browser of a PC.*

##### a. Devices

Device Type	Device Name
Home Gateway	Home Gateway0
Door	door0
Door	door1
PT-PC	PC0

##### b. Configure gateway address on **Home Gateway**

- Go to **Home Gateway** -> Config -> LAN
- IP: 192.168.25.1 / 255.255.255.0

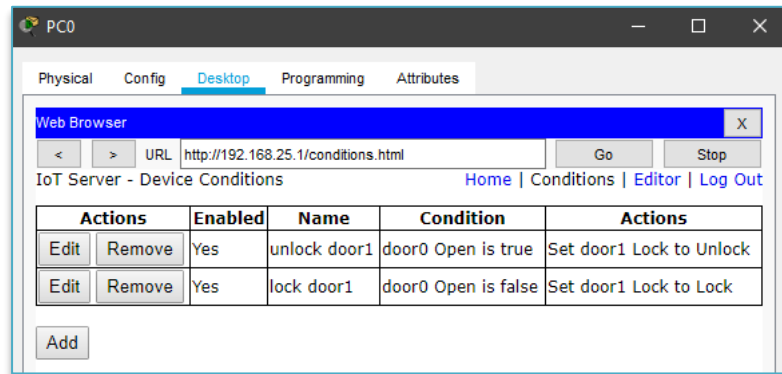
- c. In order to communicate with the **Home Gateway**, the devices have to be configured with an IP address. In PT (Packet Tracer), the **Home Gateway** Device is running a DHCP Service by default. Therefore, we can configure the other devices to use DHCP to obtain an IP automatically. Also, configure the devices to use the **Home Gateway** as the IoT Server

##### d.

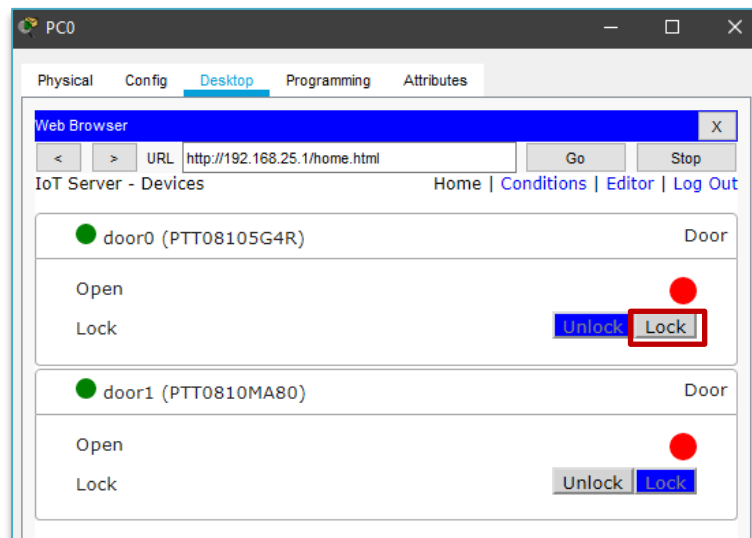
Device Name	FastEthernet0 - IP	Config->Settings
door0	via DHCP	Enable <b>Home Gateway</b>
door1	via DHCP	Enable <b>Home Gateway</b>
PC0	via DHCP	Enable <b>Home Gateway</b>

*Note: Unlike using a separate remote server as a registration server that needs credentials (extra layer of security) to communicate, using a Home Gateway does not require the credentials. This is analogous to a regular device (i.e. PC) having a wired connection to a router does not require any special credentials to work with the router. In contrast, a wireless connection would require a password for a device to be connected to an broadcasted SSID.*

- e. In order to configure the rules for the IoT devices connected to the **Home Gateway**, open a web browser on **PC0** and go to 192.168.25.1 (login with the credentials UN: **admin**, PW: **admin** / Default account)
- f. Configure the rules as shown below:



- g. Everything is configured! Let's see it in action ... 😊
- h. Open and close **door0** will cause door1 to unlock/lock (press Alt + **door0** to interact with the door)
- i. You can also lock/unlock **door0** via the web portal remotely:



### Q5: Try me! Questions

1. In Q2, what is the importance of the use of “wait()” function inside the main while loop? Is it really necessary? What could be a better way to replacing expectation of using the “wait()” function?
2. In Q3, why it is not possible to unlock, lock **door1** via the web portal remotely?
3. Open Packet Tracer -> File -> Open Samples, Try different IoT implementation samples in:
  - a. IoT/Environment
  - b. IoT/iot
  - c. IoT/IoT Devices
  - d. IoT/Programming/bluetooth music player
  - e. IoT/Programming/desktop user gui

**desktop user gui** is an example of deploying a program code into a Desktop App with a GUI which could be installed on a PC/SBC device. (Sample GUI Project: PC -> Programing -> **Desktop App - Python**)

### Summary

1. Understand the Basics of IoT
2. Configuring a simple network of things
3. Configuring a IoE (Internet of Everything) Registration Server
4. Configuring a Home Gateway
5. Try me! Questions

