SeaGlide_TEAM	Arduino with Python	Lesson #12-13 (60 minutes each)

Description:
Students will learn how Python and Arduino can be combined to create a powerful and user friendly system by becoming Software Engineers. Students will implement what they have learned in the previous lessons to set up a client server configuration, send data between Python and Arduino, and measure pressure and temperature using this configuration.

Students will be able to:

- Pass strings back and forth between Python and Arduino.
- Gain experience sending data over ethernet.
- Dissect and understand programming code.
- Use a pressure sensor to measure temperature and pressure.

Students will understand:
Arduino is a powerful, user friendly electronics platform and Python is a high-level programming language. When combined, they unlock Arduino's full potential of receiving, interpreting, and returning data. Through this lesson, the students will learn how Arduino and Python can work together to create a highly interactive communication system. By working through the activities, students will deconstruct Arduino and Pythons code to understand their structure and function.

Key Definitions & Concepts : [1]

- **Ethernet**: the standard way to connect computers on a network over a wired connection.
- **IP Address**: a unique address that identifies a device on the Internet or a local network. It allows a system to be recognized by other systems connected via the Internet protocol.
- **Integrated Development Environment (IDE)**: an application that developers use to create computer programs.
- **Media Access Control (MAC) Address**: a hardware identification number that uniquely identifies each device on a network.
- **Network**: consists of multiple devices that communicate with one another.

Standards: [Copied from: 2]
3B.DA.06 Select data collection tools and techniques to generate data sets that support a claim or communicate information.

3B.AP.12 Compare and contrast fundamental data structures and their uses.

	Background Information	

Prior Knowledge:

- Apply logic and problem solving skills dissect a program

- Pattern recognition
- Logical (step-by-step) thinking

Science Practices: [Copied from: 3]	**Core Ideas: [Copied from: 4]**	**Cross Cutting Concepts: [Copied from: 5]**
● Developing and Using Models ● Analyzing and Interpreting Data	● Optimizing the Design Solution	● Patterns ● Cause and Effect ● Structure and Function

Possible Preconceptions/Misconceptions:

Since this lesson implements topics learned in previous lessons, instructors should not expect students to experience any preconceptions. The lesson is designed in a way to help students thoroughly understand any associated concepts. In the event a student struggles with being able to translate between the languages, the instructor's guide and answer keys include enough information to help the instructor address any possible questions, misconceptions, and concerns.

Lesson Plan - 5E(+) Model

Engage: Day 1: [6]

The instructor will hand out the *Benefits of a Bilingual Brain* worksheets. Students will be engaged by watching a short video (Benefits of Being Bilingual Video). This video discusses the beneficial impact that learning multiple languages can have on the brain. The purpose of this video is to have students make a connection between learning multiple natural languages to learning multiple programming languages. Students will work individually to answer the questions related to the video while it is playing. It should take 10 minutes for the students to complete the worksheet. This includes answering questions 1-3 while watching the video and the extra time needed for answering the fourth question. The teacher should facilitate an open class discussion to review students' answers and address any misconceptions as necessary. It is important to spend most of the discussion time on the fourth question since it is the connector between learning Python and Arduino programming and since it is the introduction to the purpose of this unit plan. This section should take no more than 15 minutes to complete.

Engage: Day 2:

The instructor will hand out the *Simple Client Server Configuration Review* pre-quiz. This pre-quiz will have students answer a series of questions that is designed as a review of the material covered in Day 1 of the unit plan. The purpose of this pre-quiz is to have students recall what they have learned in the previous lesson. The goal is to have students refresh their understanding of how Arduino and Python can work as a simple client server configuration to communicate data back and forth. The instructor should allot 5 minutes for the completion of the quiz. After students have completed their quiz, the instructor will hold a class discussion reviewing the pre-quiz to ensure students have a full understanding of the concepts from the previous lesson. This section should take no more than 10 minutes to complete.

Explore: Day 1 [7]

Part I: Introduction Day 1: Simple Client Server Configuration over Ethernet

The instructor will hand out all necessary handouts and materials (listed below) for the completion of the activities for Day 1. The instructor will explain that Day 1 is allocated to understanding how to set up communication between Python software and an Arduino. This section should take 5 minutes to complete.

Materials needed for Day 1: (1 per pair)

- Arduino Uno R3 (Amazon Link)
- Arduino Ethernet Shield R3 (Amazon Link)
- Ethernet Cable
- Serial Cable
- Laptops

Part II: *Benchmark Day 1:* Simple Client Server Configuration over Ethernet

The instructor will be responsible for setting up the network to obtain an IP address for the Arduino as well as working with the students to assign a mac address. The instructor will demonstrate how the mac address and IP addresses are assigned to the Arduino and Python softwares before students set up their Arduinos. Instructions for the set up process are listed on the *Instructor's Guide*. In addition, a video explaining this process (Lesson 16: Simple Client Server Configuration over Ethernet) is available. Students will work in pairs to set up their client server configuration using the materials provided as well as the directions provided in the *Constructing a Simple Client Server Configuration over Ethernet Activity* worksheet. Students will work through the process of entering the Arduino code into their Arduino IDE and the Python code into their Python IDE. The purpose of this activity is for students to understand how Python and Arduino could be used together so that the Arduino is a server that is controlled and queried by clients. Students will answer analysis questions for both Arduino and Python to show that they have a full understanding of how the code works. This section should take the remainder of the lesson to complete (40 minutes).

Explore: Day 2 [8], [9]

Part III: Introduction Day 2: Sending and Receiving Data over Ethernet

The instructor will hand out all necessary handouts and materials (listed below) for the completion of the activities for Day 2. The instructor will explain that Day 2 is allocated to gain experience passing data back and forth between Arduino and Python with a more practical example using a pressure sensor. This section should take 5 minutes to complete.

Materials needed for Day 2: (1 per pair)

- Arduino Uno R3 (Amazon Link)
- Arduino Ethernet Shield R3 (Amazon Link)
- Adafruit BMP180 Pressure Sensor (Amazon Link)
- *Connecting Wires (4 per pair)
- Needle-nose pliers
- Ethernet Cable
- Serial Cable

- Laptops
- Python Launcher / IDE Software
- Arduino IDE Software
- Breadboard

Part IV: *Investigation Day 2:* Sending and Receiving Data over Ethernet

The instructor will be responsible for setting up the network to obtain an IP address for the Arduino as well as working with the students to assign a mac address. The instructor will demonstrate how the mac address and IP address are assigned to the arduino and Python softwares before students set up their Arduinos. Instructions for the set up process are listed on the *Instructor's Guide.* In addition, a video explaining this process (Lesson 16: Simple Client Server Configuration over Ethernet) is available. It is best if the instructor allows students to work from the same laptop from Day 1 in order to save time with the setup process. Students will continue working in the same pairs as Day 1 to set up their client server configuration as well as connecting their Arduinos to pressure sensor using a breadboard using the *Sending and Receiving Data over Ethernet* worksheet. Students will work through the process of entering the Arduino code into their Arduino IDE and the Python code into their Python IDE. The purpose of this activity is for students to gain experience using a real life example involving pressure and temperature. Students will answer analysis questions for both Arduino and Python to show that they have a full understanding of how the code works as well as use what they have learned to make a connection to their SeaGlide. This section should take the remainder of the lesson to complete (40 minutes).

Explain:

Throughout the exploration, students will engage in discussions that inquire their understanding and knowledge of the information at-hand. Instructors will be informally asking students to explain topics and relevant connections throughout the entirety of this lesson. The worksheets include "why" and "how" questions to provide the instructor with their individual explanations and to also give the students an opportunity to verbalize their understandings. The exploration also includes questions that prompt students to defend their responses.

Elaborate:

The elaboration of this lesson is the exploration section. Computer programming is currently a prominent and relevant career path that requires developers to tackle problems by reading through existing code and understanding the functions different portions of the code. The activities within this lesson unit allow students to gain experience to these programming skills. This student-designed program forces the students to think like a Software Engineer by engaging in a real-world scenario.

Evaluate:

This lesson is designed to have both informal and formal evaluations throughout its entirety. The informal evaluations occur throughout the both the engage and exploration because of the leading and open-ended questions as well as the class discussions. This allows the instructor to gauge surface-level student understanding. This is done through listening to student conversation and observing how students work through the activity worksheets. During this time, the instructor has the

ability to hear and address misconceptions or misunderstandings as necessary. The formal evaluations of this lesson is the *Constructing a Simple Client Server Configuration over Ethernet Activity* worksheet, the *Simple Client Server Configuration Review* pre-quiz, and the *Sending and Receiving Data over Ethernet* worksheet. These activities assess students on what they have learned and prompt them to provide detailed explanations to support their responses.

Enrich:

This lesson could be differentiated by converting it into a university level computer science course due to its use of multilingual programming languages and its application to real-life scenarios. Upon instructor discretion, the students could learn how to set up the communication configuration, network, and write the code to send/receive data instead of having the code provided. This type of course could be prevalent for students in a department of Information Science and Technology. They would benefit from a course like this because they would be able to go beyond just the programming aspect and learn how multiple softwares communicate with each other.

All associated documents are attached below

Reference *Annotated Bibliography* on the very last page of this packet

Worksheets for Day 1: Simple Client Server Configuration over Ethernet

Names: _____ Date: _____

Benefits of a Bilingual Brain Worksheet [6]

1. Explain the difference between compound, coordinate, and subordinate bilingual.

   Compound bilingual:

   Coordinate bilingual:

   Subordinate bilingual:

2. Define the critical period hypothesis.

3. List the three defining skills of executive function.

4. Using what you have learned from the video, connect being multilingual in natural language to being multilingual in computer languages. Detail your response.

Instructor's Guide

Simple Client Server Configuration over Ethernet [7]

Useful Links

- Lesson 16: Simple Client Server Configuration over Ethernet
- Lesson 16 Youtube Video

Goal: pass a string back and forth between Python and Arduino (if you can pass strings, you can pass anything).

It would be beneficial if the instructor takes the time to watch the instructional video to have full understanding of how the setup process works as well as the code for both Arduino and Python.

Programming Portion:

Goal: give enough information to see how the technique (pass a string back and forth between Python and Arduino) is done and apply that technique to other aspects.

Setting up the Server on the Arduino:

- How to Setup the IP Address:
    - **Load Libraries** (code lines 1-3):
        - Ethernet library
        - Ethernet UDP Library
            - Easiest way to send commands and data between Python and Arduino is using UDP
        - SPI Library

    ** These libraries should already be included in the arduino IDE, no additional installation required

    - **Identifying the Arduino to the Network** (codelines 5-11)
        - Assign mac address (line 5): in software and then connect to the ethernet from the address entered in the software because the arduino shields do not have a built-in hard-wired mac address
            - **IMPORTANT**: Assign a mac address that is **UNIQUE** to **YOUR** network
            - *Can use the mac address given in the code, IF it is NOT already on your network
        - Assign an IP Address (line 6): talk to a school network administrator to work out receiving an IP Address for the arduino
            - If you plug the ethernet cord directly to the arduino shield, it should show up on the router with a mac address and an IP address because the router automatically assigns an IP address. In this case, you will use the IP address given by the router and assign it to your arduino
        - Assign a port of communication (line 7): a port that you will talk over
            - Assign a variable and set it to a value
        - Declare a char variable (line 8): The data will be sent back and forth over a character array (variable of type character in an array so that you will be able to have multiple characters)

        - Assign a dimension (a maximum size)
    - Define a string (line 9): you want to create a string that you can store the data in once you get it
        - the data will be going back and forth between the string and the array
    - Define a variable for packet size; (line 10): Need to know the size of the packet you are sending back an forth, so a variable needs to be defined for it
    - Define an object for DUP interactions (line 11): Need to create an object that you can interact with to do the DUP interaction

- **Void Setup Loop** (code lines 13-19)
    - Turn on the serial port (line 15): Use a serial port because you may want to look at the serial monitor, and see data as it is coming in and out
    - Turn on the ethernet (line 16): need to turn it on and tell it where you will be communicating over (the mac address and the IP address)
    - Command the UDP to begin (line 17): need to initialize the UDP
    - Create a delay (line 18): initialize a delay to allow time for everything to begin

- **Void Loop** (code lines 22-50)
    - Simple client-server relationship
        - Python (client) - will ask for what it wants (in this instance a color)
            - This can be applied in different situations
        - Arduino (server) - will fulfill the request and return it to Python (will respond, “You requested *color*)
    - The remainder of the code talks about how the data is received and how the arduino fulfills the request

Write code for the client on Python (44:30 - 1:02:07):

- **Load Libraries** (code lines 1-2)
    - Same explanation as for the Arduino code
- Identify the server (lines 4) Identify the server who Python will be communicating with
- Create a socket (line 5) identify and set up the line of communication for the client
- (line 6) Set up a parameter where if Python puts in a request with Arduino and it doesn’t hear back in 1 second, it gives up (prevents it from crashing)
- (lines 8 - 37) Create a main loop that lists the commands that will be sent to Arduino and reads the received data

Names: _____ Date: _____

Constructing a Simple Client Server Configuration over Ethernet Activity [8]

Introduction: Arduino is a powerful, user friendly electronics platform that is used internationally. An Arduino board's main function is to be able to read inputs and convert them into outputs. A user is able to communicate with the arduino using the Arduino IDE software to assign commands to the Arduino. However, Arduino lacks the ability to display the data it reads. Connecting the Arduino to a full programming language, like Python, allows the user to unlock the full potential of the Arduino while also being able to display the data in various styles. Today, you and your partner will become Software Engineers and you will help a client test out a new method of communication between Arduino and Python. You will work through the process of setting up a client server configuration over ethernet using strings. Then, you will explain the supporting codes for Arduino and Python and analyze your process to help the client understand your findings.

Directions: First, follow the steps below to setup the client server configuration between Arduino and the computer. Then, answer the analysis questions using the provided code.

Materials:

- Arduino Uno
- Arduino Ethernet Shield
- Network (or router)
- Ethernet Cable
- Serial Cable
- Laptop
- Python Launcher / IDE Software
- Arduino IDE Software

Setup: use *Figure 1* to help with your setup process

- Connect the Arduino to the Arduino Ethernet Shield to create a single unit that contains a serial port and an ethernet port
- Connect one end of the ethernet cable into the ethernet port on the Arduino Ethernet Shield
- Connect to the network using the ethernet cable and the ethernet port on the Arduino Ethernet Shield
- Connect one end of the serial cable to the the serial port on the Arduino Uno
- Connect the opposite end of the serial cable to the computer

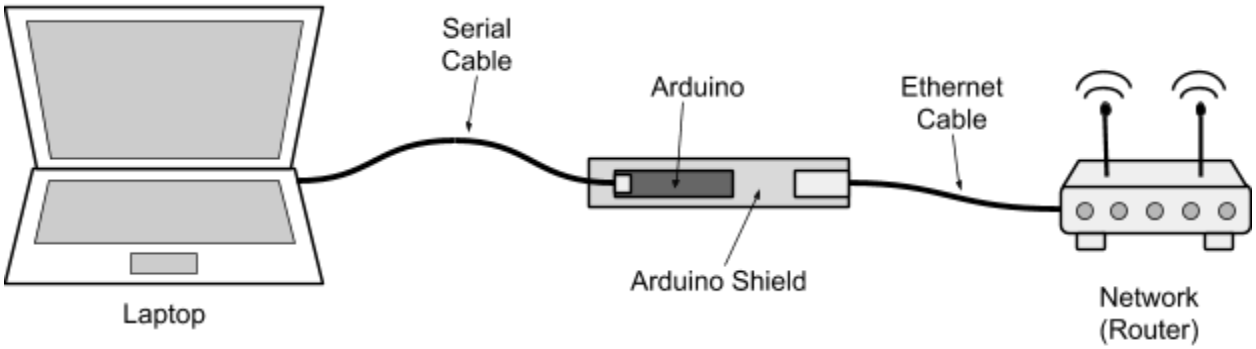

Figure 1: Client Server Configuration Setup Schematic

Python Client Code:

```
from socket import *
import time

address = ( '10.1.15.243', 5000) #Defind who you are talking to (must match arduino IP and port)
client_socket = socket(AF_INET, SOCK_DGRAM) #Set Up the Socket
client_socket.settimeout(1) #only wait 1 second for a resonse

while(1): #Main Loop

    data = "Blue" #Set data to Blue Command
    client_socket.sendto(data, address) #send command to arduino
    try:
        rec_data, addr = client_socket.recvfrom(2048) #Read response from arduino
        print rec_data #Print the response from Arduino
    except:
        pass

    time.sleep(2) #delay before sending next command

    data = "Red" #Set data to Blue Command
    client_socket.sendto(data, address) #send command to arduino
    try:
        rec_data, addr = client_socket.recvfrom(2048) #Read response from arduino
        print rec_data #Print the response from Arduino
    except:
        pass

    time.sleep(2) #delay before sending next command

    data = "Green" #Set data to Blue Command
    client_socket.sendto(data, address) #send command to arduino
    try:
        rec_data, addr = client_socket.recvfrom(2048) #Read response from arduino
        print rec_data #Print the response from Arduino
    except:
        pass
    time.sleep(2) #delay before sending next command
```

Python Analysis Questions

1. Explain the importance of the set up process you conducted between Python and Arduino. Use evidence to support your response.

2. Explain how Python is communicating with Arduino. Use evidence from both the Python client code and the Arduino server code to support your response.

3. Explain what is happening at the following lines of the Python code:

   Lines 1-2:

   Lines 4-6:

   Lines 8-16:

   Line 18:

Arduino Server Code:

```
#include <Ethernet.h> //Load Ethernet Library
#include <EthernetUdp.h> //Load UDP Library
#include <SPI.h> //Load the SPI Library

byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xEE}; //Assign a mac address
IPAddress ip(10, 1, 15, 243); //Assign my IP adress
unsigned int localPort = 5000; //Assign a Port to talk over
char packetBuffer[UDP_TX_PACKET_MAX_SIZE];
String datReq; //String for our data
int packetSize; //Size of Packet
EthernetUDP Udp; //Define UDP Object

void setup() {

Serial.begin(9600); //Turn on Serial Port
Ethernet.begin(mac, ip); //Initialize Ethernet
Udp.begin(localPort); //Initialize Udp
delay(1500); //delay
}

void loop() {

  packetSize = Udp.parsePacket(); //Read theh packetSize

  if(packetSize>0){ //Check to see if a request is present

  Udp.read(packetBuffer, UDP_TX_PACKET_MAX_SIZE); //Reading the data request on the Udp
  String datReq(packetBuffer); //Convert packetBuffer array to string datReq

  if (datReq =="Red") { //See if Red was requested

    Udp.beginPacket(Udp.remoteIP(), Udp.remotePort());  //Initialize Packet send
    Udp.print("You are Asking for Red"); //Send string back to client
    Udp.endPacket(); //Packet has been sent
  }
   if (datReq =="Green") { //See if Green was requested

    Udp.beginPacket(Udp.remoteIP(), Udp.remotePort());  //Initialize Packet send
    Udp.print("You are Asking for Green"); //Send string back to client
    Udp.endPacket(); //Packet has been sent
   }
    if (datReq =="Blue") { //See if Red was requested

    Udp.beginPacket(Udp.remoteIP(), Udp.remotePort());  //Initialize Packet send
    Udp.print("You are Asking for Blue"); //Send string back to client
    Udp.endPacket(); //Packet has been sent
    }
  }
  memset(packetBuffer, 0, UDP_TX_PACKET_MAX_SIZE);
}
```

Arduino Analysis Questions

1. For the given lines of the Arduino code, answer each of the following questions:
    a. Define the lines of the corresponding Python code
    b. Explain the correspondence between Python and Arduino

    Lines 15-18:

    Lines 25-28:

    Lines 30-35:

    Lines 36-41:

    Lines 42-47:

Worksheets for Day 2: Sending and Receiving Data over Ethernet

Names: _____ Date: _____

Simple Client Configuration Review

1. In a client server configuration between Python and Arduino, identify which one is the client and which one is the server.

2. Explain what happens when when Python sends Arduino a request.Make sure to be as detailed as possible.

3. For the given lines of Python code below, explain the communication occurring between Python and Arduino.

```
data = "Blue" #Set data to Blue Command
client_socket.sendto(data, address) #send command to arduino
try:
    rec_data, addr = client_socket.recvfrom(2048) #Read response from arduino
    print rec_data #Print the response from Arduino
except:
    pass

time.sleep(2) #delay before sending next command

data = "Red" #Set data to Blue Command
client_socket.sendto(data, address) #send command to arduino
try:
    rec_data, addr = client_socket.recvfrom(2048) #Read response from arduino
    print rec_data #Print the response from Arduino
except:
    pass

time.sleep(2) #delay before sending next command
```

Instructor's Guide

Sending and Receiving Data over Ethernet [9]

Useful Links:

- Lesson 17: Sending and Receiving Data Over Ethernet
- Lesson 9: Measuring Pressure and Temperature with the BMP180 Sensor
    - For directions on how to connect a sensor to an arduino
- Lesson 17 Youtube Video

Goals:

- Gain further experience passing strings back and forth to show very basic Server on Arduino, and Python as the client.
- Shows a more practical example using a pressure sensor.

It would be beneficial if the instructor takes the time to watch the instructional video to have full understanding of how the setup process works as well as the code for both Arduino and Python.

Arduino Code Development (in the Arduino IDE)
(repetitive from lesson 16)

- Load libraries (lines 1-3)
    - Ethernet library
    - Ethernet UDP library
    - SPI Library
- Load Libraries for the pressure sensor
    - Wire library (line 5)
    - Pressure/Temperature sensor library Adafruit BMP 085 (line 6) (have to install the adafruit library because it does not come with the Arduino IDE: https://learn.adafruit.com/bmp085/using-the-bmp085)
      [Click the "Download the Adafruit_BMP085 Arduino Library" large green box. This will download as a zip folder. Open the zip folder, and then drag and drop the contents on your desktop. You want the contents of the zip folder, not the zip folder itself. Rename the folder you dropped to your desktop "adafruitBMP180". Now you need to drag and drop this folder into your arduino library folder. To find your arduino library folder, in the arduino IDE window, look in file, preferences. A window should pop open, and it should show you where your arduino sketchbook folder is. Drop your adafruitBMP180 folder into the Library folder of your arduino sketchbook folder. If this is not perfectly clear, watch the video above and you can watch me do it step-by-step. Once your adafruitBMP180 folder is in your arduino library folder, you are ready to start writing your code. You need to kill your arduino IDE window and reopen it for it to find your new library.]
    - Create a sensor object (line 7): will use to interact with the sensor
    - Declare variables for temperature and pressure (lines 9 - 11):
        - These variables will be used to measure temperature (in Celsius and Fahrenheit) and pressure
    - Ethernet Setup (lines 13-20): Refer to Day 1 Instructor's guide
        - Assign the mac address
        - Assign the IP address

        - Assign a communication port
        - Create an array to store the data
        - Declare a string for the data
    - Void Setup - turning everything on
        - Turn on the serial port (line 23)
        - Start the ether (line 24)
            - Tell it the mac address and IP address
        - Initialize the UDP (line 25)
            - Tell it which port you will be communicating over
        - Set a delay to allow time for the arduino to get going (line 28)
        - Turn on the adafruit (line 27)
    - Void loop
        - Has the arduino wait and check for command requests, determines whether the request is for temperature or pressure

<u>Python Code Development</u>

- Load libraries
    - Socket library: create and use sockets to talk over ethernet
- Import time for delays
- Identify who python will be communicating with using the IP address
- Create a socket: in which the client and server will communicate
- While loops
    - Types of requests Python will make to the arduino

Names: _____ Date: _____

Sending and Receiving Data over Ethernet [10][11]

Introduction: Arduino is a powerful, user friendly electronics platform that is used internationally. An Arduino board's main function is to be able to read inputs and convert them into outputs. A user is able to communicate with the arduino using the Arduino IDE software to assign commands to the Arduino. However, Arduino lacks in the ability to display the data it reads. Connecting the Arduino to a full programming language, like Python, allows the user to unlock the full potential of the Arduino while also being able to display the data in various styles. Today, you and your partner will continue your position as Software Engineers, and you will help a client test out a new method of communication between Arduino and Python. Using the skills you learned in the first portion of the lesson, you will apply a more practical approach by setting up your Arduino to measure temperature and pressure. Then you will explain the supporting codes for Arduino and Python and analyze your process to help the client understand your findings.

Directions: First, follow the steps below to setup the client server configuration between Arduino and the computer as well as the pressure sensor. Then, answer the analysis questions using the provided code.

Materials

- Arduino Uno
- Arduino Ethernet Shield
- Network (or router)
- Ethernet Cable
- Adafruit BMP180 Pressure Sensor
- Connecting Wires (4)
- Needle-nose pliers
- Serial Cable
- Laptop
- Python Launcher / IDE Software
- Arduino IDE Software
- Breadboard

Setup:

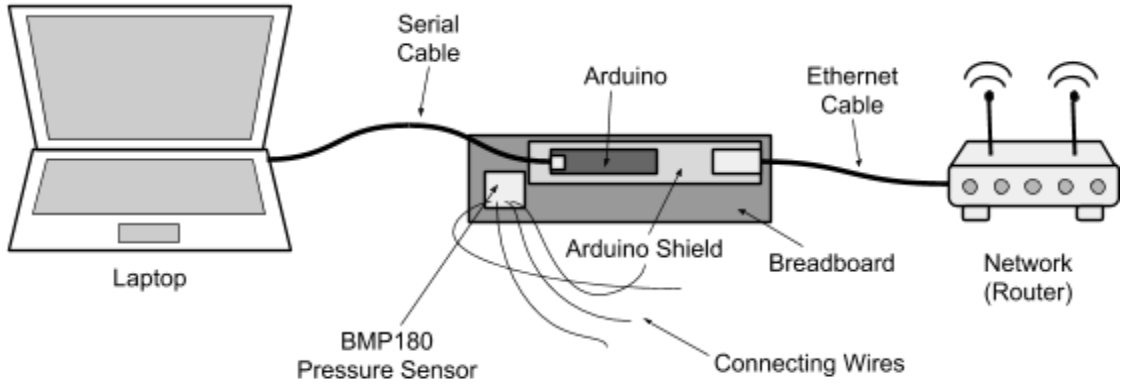

Figure 1: Client Server Configuration Setup Schematic with an Adafruit BMP180 Pressure Sensor

Setup Continued:

- Connect the Arduino to the Arduino Ethernet Shield to create a single unit that contains a serial port and an ethernet port and connect it to the breadboard
- Connect the Adafruit BMP180 Pressure sensor to the breadboard
- Attach the connecting wires using the table below
- Connect one end of the ethernet cable into the ethernet port on the Arduino Ethernet Shield
- Connect to the network using the ethernet cable and the ethernet port on the Arduino Ethernet Shield
- Connect one end of the serial cable to the the serial port on the Arduino Uno
- Connect the opposite end of the serial cable to the computer

Connecting Up the BMP180 Pressure and Temperature Sensor [11]	
BMP180 Pin	Arduino Pin
Vin	5V
GND	GND
SCL	A5
SDA	A4

Python Client Code

```
from socket import *
import time

address= ( '10.1.15.243', 5000) #define server IP and port
client_socket =socket(AF_INET, SOCK_DGRAM) #Set up the Socket
client_socket.settimeout(1) #Only wait 1 second for a response

while(1):

    data = "Temperature" #Set data request to Temperature

    client_socket.sendto( data, address) #Send the data request

    try:

        rec_data, addr = client_socket.recvfrom(2048) #Read response from arduino
        temp = float(rec_data) #Convert string rec_data to float temp
        print "The Measured Temperature is ", temp, " degrees F." # Print the result

    except:
        pass

    time.sleep(2) #delay before sending next command

    data = "Pressure" #Set data request to Pressure

    client_socket.sendto( data, address) #Send the data request

    try:

        rec_data, addr = client_socket.recvfrom(2048) #Read response from arduino
        Pressure = float(rec_data) #Convert string rec_data to float temp
        print "The Measured Pressure is ", Pressure, " Pa." # Print the result

    except:
        pass

    time.sleep(2) #delay before sending next command

    print ""
```

Python Analysis Questions

```
try:

    rec_data, addr = client_socket.recvfrom(2048) #Read response from arduino
    temp = float(rec_data) #Convert string rec_data to float temp
    print "The Measured Temperature is ", temp, " degrees F." # Print the result

except:
    pass
```

1. Why are exception handlings (try and except) used in Python? How is it used in this set up? An example of an exception handling is included above.

2. What role does Python play in this client server configuration? Explain the tasks Python should complete.

3. Explain what is happening in the lines of code pictured below.

```
while(1):

    data = "Temperature" #Set data request to Temperature

    client_socket.sendto( data, address) #Send the data request

    try:

        rec_data, addr = client_socket.recvfrom(2048) #Read response from arduino
        temp = float(rec_data) #Convert string rec_data to float temp
        print "The Measured Temperature is ", temp, " degrees F." # Print the result

    except:
        pass

    time.sleep(2) #delay before sending next command
```

Arduino Server Code

```
#include <Ethernet.h> //Load Ethernet Library
#include <EthernetUdp.h> //Load the Udp Library
#include <SPI.h> //Load SPI Library

#include "Wire.h" //imports the wire library
#include "Adafruit_BMP085.h" // import the Pressure/Temperature sensor library
Adafruit_BMP085 mySensor; //Create a sensor object

float tempC; //Declare variable for Temp in C
float tempF; //Declare variable for Temp in F
float Pressure; //Declare a variable for Pressure

byte mac[] ={ 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xEE}; //Assign mac address
IPAddress ip(10, 1, 15, 243); //Assign the IP Adress
unsigned int localPort = 5000; // Assign a port to talk over
char packetBuffer[UDP_TX_PACKET_MAX_SIZE]; //dimensian a char array to hold our data packet
String datReq; //String for our data
int packetSize; //Size of the packet
EthernetUDP Udp; // Create a UDP Object

void setup() {

  Serial.begin(9600); //Initialize Serial Port
  Ethernet.begin( mac, ip); //Inialize the Ethernet
  Udp.begin(localPort); //Initialize Udp
  delay(1500); //delay
  mySensor.begin(); //initialize pressure-temp sensor


}

void loop() {

  packetSize =Udp.parsePacket(); //Reads the packet size

  if(packetSize>0) { //if packetSize is >0, that means someone has sent a request

    Udp.read(packetBuffer, UDP_TX_PACKET_MAX_SIZE); //Read the data request
    String datReq(packetBuffer); //Convert char array packetBuffer into a string called datReq

  if(packetSize>0) { //if packetSize is >0, that means someone has sent a request

    Udp.read(packetBuffer, UDP_TX_PACKET_MAX_SIZE); //Read the data request
    String datReq(packetBuffer); //Convert char array packetBuffer into a string called datReq

    if (datReq =="Temperature") { //Do the following if Temperature is requested

      tempC = mySensor.readTemperature(); //Read the temperature
      tempF = tempC*1.8 + 32; //Convert temp to F

      Udp.beginPacket(Udp.remoteIP(), Udp.remotePort()); //Initialize packet send
      Udp.print(tempF); //Send the temperature data
      Udp.endPacket(); //End the packet

    }

    if (datReq== "Pressure") { //Do the following if Pressure is requested

      Pressure=mySensor.readPressure(); //read the pressure

        Udp.beginPacket(Udp.remoteIP(), Udp.remotePort()); //Initialize packet send
        Udp.print(Pressure); //Send the Pressure data
        Udp.endPacket(); //End the packet

    }

  }
  memset(packetBuffer, 0, UDP_TX_PACKET_MAX_SIZE); //clear out the packetBuffer array
}
```

Arduino Analysis Questions

1. In the void loop, there are nested conditional statements.

    a. How does the computer know if a user has sent a request?

    b. Explain what is done if temperature is requested.

    c. How are the actions done in part B different from the actions done if pressure is requested?

2. Where does the temperature and pressure data get sent to?

3. Why does the packetBuffer array have to be cleared of all data?

BONUS: How could this type of technology be used in a SeaGlide?

Answer Keys for Day 1: Simple Client Server Configuration over Ethernet

Name: _____ANSWER KEY_____ Date: _____

Benefits of a Bilingual Brain Worksheet [6]

1. Explain the difference between compound, coordinate, and subordinate bilingual.

   Compound bilingual: developing two linguistic codes simultaneously with a single set of concepts.

   Coordinate bilingual: learning a secondary language by working with two sets of concepts.

   Subordinate bilingual: learning a secondary language by filtering it through a primary language.

2. Define the critical period hypothesis.

   Children learn languages more easily because the plasticity of their developing brain lets them use both hemispheres in language acquisition. While most adults' language is lateralized to one hemisphere, usually to the left, childrens' brain plasticity allows for utilization of both hemispheres.

3. List the three defining skills of executive function.

   Problem solving

   Switching between tasks

   Focusing while filtering out irrelevant information

4. Using what you have learned from the video, connect being multilingual in natural language to being multilingual in programming languages. Detail your response.

   The benefits of being multilingual in natural languages can carry over to being multilingual in programming languages. Being multilingual in programming languages allows an individual to improve their executive function, use more brain activity when engaging in programing, and build the skill of breaking a problem down into a stepwise solution.

Name: _____ANSWER KEY_____ Date: _____

Constructing a Simple Client Server Configuration over Ethernet Activity

Python Analysis Questions

1. Explain the importance of the set up process you conducted between Python and Arduino. Use evidence to support your response.

   The set up process ensures that Python and Arduino are connected to the same network, therefore, allowing the to communicate back and forth with each other. If Arduino and Python were connected to separate networks, the communication between the two would not be able to occur because Python could send a request, but the Arduino would not be able to receive it.

2. Explain how Python is communicating with Arduino. Use evidence from both the Python client code and the Arduino server code to support your response.

   Python is communicating with Arduino by using a local port and an IP Address. Line 4 of the Python code and line 7 of the Arduino code show that Python is talking to Arduino by using matching IP Addresses over local port 5000.

3. Explain what is happening at the following lines of code:

   Lines 1-2:

   Load libraries to be able to implement network communication and time delays in the code.

   Lines 4-6:

   Tell Python that it is communicating with Arduino over a local port. Setting up the socket to identify the network in which Python and Arduino are communicating over. Line 6 tells Python to wait 1 second for a response from Python.

   Lines 8-16:

   A while loop where Python sends commands to Arduino. Lines 10-16 is specifically for the color blue. Python sends a command to Arduino requesting a string identifying the color blue. Python then reads the response that Arduino returns and prints the response to the user.

   Line 18:

   Line 18 has Python delay for 2 seconds before moving on to the next command.

Arduino Analysis Questions

1. For the given lines of the Arduino code, answer each of the following questions:
    a. Define the lines of the corresponding Python code
    b. Explain the correspondence between Python and Arduino

Lines 15-18

a. Lines 4-6
b. Initializes the communication port between Python and Arduino, the ethernet, UDP, and gives a 15 second delay to allow time for everything to set up.

Lines 25-28

a. Lines 10-11
b. Checks to see if a request has been made from Python by checking the UDP size. If the size is greater than 0, then a request has been made. Line 28 converts the array to a string request.

Lines 30-35

a. Lines 20-26
b. Python sends a data request for the color red. Arduino receives the request, converts to string data request, sends a string back to Python and confirms that the data has been sent.

Lines 36-41

a. Lines 30-36
b. Python sends a data request for the color green. Arduino receives the request, converts to string data request, sends a string back to Python and confirms that the data has been sent.

Lines 42-47

a. Lines 40-46
b. Python sends a data request for the color blue. Arduino receives the request, converts to string data request, sends a string back to Python and confirms that the data has been sent.

Answer Keys for Day 2: Sending and Receiving Data over Ethernet

Names: _____ Date: _____

Simple Client Configuration Review

1. In a client server configuration between Python and Arduino, identify which one is the client and which one is the server.

   Python is the client and Arduino is the server.

2. Explain what happens when when Python sends Arduino a request. Make sure to be as detailed as possible.

   When Python sends a request, the Arduino UDP reads packet size to be greater than 0 which allows it to know it has received a request. The Arduino fulfills the request by reading the packet, identifying the color, and returning a string that read “You are Asking for [color]”. Python then receives the returned data from Arduino and prints it.

3. For the given lines of Python code below, explain the communication occurring between Python and Arduino.

```
data = "Blue" #Set data to Blue Command
client_socket.sendto(data, address) #send command to arduino
try:
    rec_data, addr = client_socket.recvfrom(2048) #Read response from arduino
    print rec_data #Print the response from Arduino
except:
    pass

time.sleep(2) #delay before sending next command

data = "Red" #Set data to Blue Command
client_socket.sendto(data, address) #send command to arduino
try:
    rec_data, addr = client_socket.recvfrom(2048) #Read response from arduino
    print rec_data #Print the response from Arduino
except:
    pass

time.sleep(2) #delay before sending next command
```

   Python sets the data to “blue” and sends a command request to Arduino for the “blue” data set. Once a response is returned, it reads the response from Arduino and prints it.

Names: _____ Date: _____

Sending and Receiving Data over Ethernet [10][11]

Python Analysis Questions

```
try:

    rec_data, addr = client_socket.recvfrom(2048) #Read response from arduino
    temp = float(rec_data) #Convert string rec_data to float temp
    print "The Measured Temperature is ", temp, " degrees F." # Print the result

except:
    pass
```

1. Why are exception handlings (try and except) used in Python? How is it used in this set up? (An example of an exception handling is included above.)

   Exception handlings are used in Python to handle possible errors. The error exceptions are caught in the “try” blocks and are handled in the “except” blocks.

   Exception handlings are used in the set up when dealing receiving the response from the Arduino after send the command request.

2. What role does Python play in this client server configuration? Explain the tasks Python should complete.

   Python is the server in the client server configuration. Python is in charge of sending the command request to Arduino, receiving the responding data, and printing the response that is received.

3. Explain what is happening in the lines of code pictured below.

```
while(1):

    data = "Temperature" #Set data request to Temperature

    client_socket.sendto( data, address) #Send the data request

    try:

        rec_data, addr = client_socket.recvfrom(2048) #Read response from arduino
        temp = float(rec_data) #Convert string rec_data to float temp
        print "The Measured Temperature is ", temp, " degrees F." # Print the result

    except:
        pass

    time.sleep(2) #delay before sending next command

    data = "Pressure" #Set data request to Pressure

    client_socket.sendto( data, address) #Send the data request

    try:

        rec_data, addr = client_socket.recvfrom(2048) #Read response from arduino
        Pressure = float(rec_data) #Convert string rec_data to float temp
        print "The Measured Pressure is ", Pressure, " Pa." # Print the result

    except:
        pass

    time.sleep(2) #delay before sending next command

    print ""
```

In the code, Python sends a request to Arduino for temperature to a specific port. Python waits for a response. When the response is received, Python reads it and converts the data into a float. Python then prints the temperature in the form of a string. Python then waits 2 seconds before moving onto the next command. Python sends a request to Arduino for pressure to a specific port. Python waits for a response. When the response is received, Python reads it and converts the data into a float. Python then prints the pressure in the form of a string. Python then waits 2 seconds before moving onto the next command.

Arduino Analysis Questions

1. In the void loop, there are nested conditional statements.

    a. How does the computer know if a user has sent a request?

    The computer knows if a user has sent a request by checking the packet size. If the packet size is greater than 0, then it knows it has received a request.

    b. Explain what is done if temperature is requested.

    If temperature is requested, Arduino measured the temperature using the pressure sensor and sends the data received back to Python.

    c. How are the actions done in part B different from the actions done if pressure is requested?

    The actions are different because, if pressure is requested, Arduino measures the pressure using the pressure sensor instead of temperature.

2. Where does the temperature and pressure data get sent to?

    The data gets sent back to Python, the client.

3. Why does the packetBuffer array have to be cleared of all data?

    The packetBuffer array has to be cleared because it is an array. If it wasn't cleared when new data is requested, it would overwrite the previous data in the packetBuffer array. This could be an issue because old data could remain in the array that has not been overwritten. Clearing the data prevents mixing old data with new data in the array.

BONUS: How could this type of technology be used in a SeaGlider?

Student responses may vary based off of what they recall from the Sensor Technology & Programming module.

Annotated Bibliography

[1] The Tech Terms Computer Dictionary. (n.d.). Retrieved from https://techterms.com/

This website was used for adaptation within the Arduino with Python lesson plan as part of the Computer Science module. This reference aided in the completion of providing definitions for the key concepts and definitions sections and for associated worksheets. The key concepts and definitions were adapted based on the grade and activities at-hand.

[2] Standards Aligned System. (n.d.). Retrieved from https://www.pdesas.org/

This website was used in each lesson in the Computer Science module to select proper Pennsylvania State standards, which are based in Common Core, that each lesson is centered around.

[3] Nsta. (n.d.). Science and Engineering Practices. Retrieved from https://ngss.nsta.org/PracticesFull.aspx

This website used in every lesson in the Computer Science module to find Standards for Scientific Practices that are applicable in each lesson.

[4] Nsta. (n.d.). Disciplinary Core Ideas. Retrieved from https://ngss.nsta.org/DisciplinaryCoreIdeasTop.aspx

This website was used in each lesson in the Computer Science module to select appropriate disciplinary core ideas set forth by the NSTA that are at the center of each lesson.

[5] Nsta. (n.d.). Crosscutting Concepts. Retrieved from https://ngss.nsta.org/CrosscuttingConceptsFull.aspx

This website was used in each lesson in the Computer Science module to selecting appropriate crosscutting concepts set forth by the NSTA that apply to each Python lesson.

[6] TED-Ed. (2015, June 23). The benefits of a bilingual brain - Mia Nacamulli. Retrieved from https://www.youtube.com/watch?v=MMmOLN5zBLY

This video was used for adaptation within the Arduino with Python lesson plan as part of the Computer Science module. Questions were developed based on this video for students to answer within the engagement portion of the lesson.

[7] McWhorter, P. (n.d.). Python with Arduino Lesson 16: Simple Client Server Configuration Over Ethernet. Retrieved from http://www.toptechboy.com/tutorial/python-with-arduino-lesson-16-simple-client-server-configuration-over-ethernet/

This website was used for adaptation within the Arduino with Python lesson plan as part of the Computer Science module for students to learn about how Python and Arduino can be used together. This reference aided in the completion of analysis questioning through adaptation. The programming codes for both Python and Arduino were excerpted directly from the website.

[8] McWhorter, P. (2015, April 03). Arduino with Python LESSON 16: Simple Client Server Model Over Ethernet. Retrieved from https://www.youtube.com/watch?v=TWGV47WrKmQ

This video was used for adaptation within the Arduino with Python lesson plan as part of the Computer Science module to be used as an instructor's guide. The video was adapted into a guide explaining the setup process as well as the written code used in the lesson.

[9] McWhorter, P. (n.d.). Python with Arduino Lesson 17: Sending and Receiving Data Over Ethernet. Retrieved from

http://www.toptechboy.com/tutorial/python-with-arduino-lesson-17-sending-and-receiving-data-over-ethernet/

This website was used for adaptation within the Arduino with Python lesson plan as part of the Computer Science module for students to learn about how Python and Arduino can be used together. This reference aided in the completion of analysis questioning through adaptation. The programming codes for both Python and Arduino were excerpted directly from the website.

[10] McWhorter, P. (2015, April 06). Arduino with Python LESSON 17: Transferring Data over Ethernet UDP. Retrieved from https://www.youtube.com/watch?v=eoiuhPxCcM4

This video was used for adaptation within the Arduino with Python lesson plan as part of the Computer Science module to be used as an instructor's guide. The video was adapted into a guide explaining the setup process as well as the written code used in the lesson.

[11] McWhorter, P. (n.d.). Python with Arduino Lesson 9: Measuring Pressure and Temperature with the BMP180 Sensor. Retrieved from http://www.toptechboy.com/tutorial/python-with-arduino-lesson-9-measuring-pressure-and-temperature-with-the-bmp180-sensor/

This website was used for adaptation and excerption within the Arduino with Python lesson plan as part of the Computer Science module for students to learn about how to setup the Arduino pressure sensor. The connection table was excerpted directly from the website.