Internet of Things Things

kulikows@cs.rutgers.edu
mcgrew@cs.rutgers.edu
Week 1: Definitions, mapping out the course
Week 2: Composition of an IOT Thing. Programming it.
        The IOT before the “I”: Apollo AGC (1961-71)
Week 3: Introduction to our processor: The Arduino
        Distribution of hardware WHICH I WANT BACK
Week 4: Security of IOT things.
Week 5: Getting used to the Arduino and building devices
Week 6: Show-offs; turn in your code
        Uses of IOTs: medicine, "smart cities", political oppression
Week 7: Research description: you pick an IOT, and research it for
        presentation to the class
Week 8: Security of IOT things #2: security, and liability
Week 9: Presentations of research projects part 1
Week 10: Presentations of research projects part 2
Arduino software install

https://www.arduino.cc/en/Main/Software

… select and install for whatever OS you’ve got.

Power: either power cable OR USB cable.

Program: runs most recently downloaded program… Forever

Check out

THE CIRCUIT:

Parts:
- CIRC-01 Breadboard Sheet x1
- 330 Ohm Resistor
  Orange-Orange-Brown x1
- 2 Pin Header x4
- 5mm Yellow LED x1
- Wire

Schematic

Arduino
pin 13

LED
(light emitting diode)

resistor (330 ohm)
(orange-orange-brown)

gnd
(ground [:) )

The Internet

download:
breadboard layout sheet
http://www.arduino.org/BDL015

download:
assembly video
http://www.arduino.org/VIDE01
CIRC-01
Getting Started (Blinking LED)

http://ardx.org/CIRC01
Circuit design – one led

Power
(arduino digital pin)

Schematic

Arduino pin 13
longer lead
LED
(light emitting diode)
resistor (330ohm)
(orange-orange-brown)
gnd
(ground) (-)

To limit the current through the LED
**Code** (no need to type everything in just click)

**File > Examples > Digital > Blink**
(example from the great arduino.cc site, check it out for other ideas)

```c
/* Blink
* Turns on an LED on for one second, then off for one second, repeated.
* Created 1 June 2005 by David Cuartielles
* http://arduino.cc/en/Tutorial/Blink
* based on an orginal by H. Barragan for the Wiring i/o board
*/

int ledPin = 13; // LED connected to digital pin 13

// The setup() method runs once, when the sketch starts
void setup() { // initialize the digital pin as an output:
    pinMode(ledPin, OUTPUT);
}

// the loop() method runs over and over again,
// as long as the Arduino has power
void loop()
{
    digitalWrite(ledPin, HIGH);  // set the LED on
    delay(1000); // wait for a second
    digitalWrite(ledPin, LOW); // set the LED off
    delay(1000); // wait for a second
}
```
Networking: How things work

All internet devices have an “IP Address” that looks like 198.1.100.5.

All devices that want to talk to anything off its immediate network must have a GATEWAY to pass traffic on. Gateways generally can figure out how to get traffic from there to there – or at least from here to the next gateway. Gateways tell each other who can send what to whom in a process that, thankfully, we never have to know about.

So all OUR device needs to know is its IP address, and its gateway. (Why does it need to know its own IP?)
Thermister

THE CIRCUIT:

Parts:
- CIRC-10 Breadboard Sheet x1
- TMP36 Temperature Sensor x1
- 2 Pin Header x4
- Wire x1

Schematic:
- Arduino analog pin 0
- +5 volts
- TMP36 (precision temperature sensor)
- gnd (ground) (-)

The Internet:
- Download:
  breadboard layout sheet
  http://ardx.org/BBL510S
- View:
  assembly video
  http://ardx.org/VIDE10
Networking: How things work

All internet devices have an “IP Address” that looks like 198.1.100.5.

All devices that want to talk to anything off its immediate network must have a **GATEWAY** to pass traffic on. Gateways generally can figure out how to get traffic from there to there – or at least from here to the next gateway. Gateways tell each other who can send what to whom in a process that, thankfully, we never have to know about.

So all **OUR** device needs to know is its IP address, and its gateway. (Why does it need to know its own IP?)

…so it knows when incoming traffic is for itself. (All other traffic it ignores.)
## My house to somewhere inside yahoo.com

<table>
<thead>
<tr>
<th>IP Address</th>
<th>Latency (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.1.1 (192.168.1.1)</td>
<td>1.562 ms</td>
</tr>
<tr>
<td>bd1.eas-cbr2.atw-eas.pa.cable.rcn.net (10.48.192.1)</td>
<td>21.354 ms</td>
</tr>
<tr>
<td>bdle30-sub212.aggr2.phdl.pa.rcn.net (207.172.196.227)</td>
<td>13.677 ms</td>
</tr>
<tr>
<td>bdle1.core1.phdl.pa.rcn.net (207.172.15.34)</td>
<td>17.109 ms</td>
</tr>
<tr>
<td>bdle7.core1.phdl.pa.rcn.net (207.172.15.50)</td>
<td>17.787 ms</td>
</tr>
<tr>
<td>bdle6.core2.phdl.pa.rcn.net (207.172.15.35)</td>
<td>17.636 ms</td>
</tr>
<tr>
<td>207.172.19.104 (207.172.19.104)</td>
<td>15.728 ms</td>
</tr>
<tr>
<td>hge0-4-0-0.border1.eqnx.va.rcn.net (207.172.19.88)</td>
<td>30.480 ms</td>
</tr>
<tr>
<td>g2-12-bas2.dce.yahoo.com (206.126.236.2)</td>
<td>16.768 ms</td>
</tr>
<tr>
<td>ae-4.pat2.che.yahoo.com (216.115.101.145)</td>
<td>46.336 ms</td>
</tr>
<tr>
<td>ae-7.pat1.nez.yahoo.com (216.115.104.124)</td>
<td>43.754 ms</td>
</tr>
<tr>
<td>et-19-1-0.msr1.ne1.yahoo.com (216.115.105.27)</td>
<td>44.119 ms</td>
</tr>
<tr>
<td>et-19-1-0.msr2.ne1.yahoo.com (216.115.105.181)</td>
<td>50.176 ms</td>
</tr>
<tr>
<td>et-19-1-0.clr1-a-gdc.ne1.yahoo.com (98.138.97.75)</td>
<td>43.882 ms</td>
</tr>
<tr>
<td>et-19-1-0.clr2-a-gdc.ne1.yahoo.com (98.138.97.63)</td>
<td>48.511 ms</td>
</tr>
<tr>
<td>et-17-1.fab3-1-gdc.ne1.yahoo.com (98.138.0.77)</td>
<td>47.329 ms</td>
</tr>
<tr>
<td>po-15.bas1-7-prd.ne1.yahoo.com (98.138.240.16)</td>
<td>59.481 ms</td>
</tr>
<tr>
<td>po-13.bas2-7-prd.ne1.yahoo.com (98.138.240.28)</td>
<td>50.384 ms</td>
</tr>
</tbody>
</table>
traceroute to nypost.com (192.0.79.33), 64 hops max, 52 byte packets
1 192.168.1.1 (192.168.1.1) 2.155 ms 0.922 ms 0.911 ms
2 bdl1.eas-cbr2.atw-eas.pa.cable.rcn.net (10.48.192.1) 8.493 ms 15.667 ms 9.983 ms
3 bdle30-sub212.aggr2.phdl.pa.rcn.net (207.172.196.227) 15.912 ms 12.018 ms 11.563 ms
4 bdle8.core2.phdl.pa.rcn.net (207.172.15.51) 16.462 ms
   bdle7.core1.phdl.pa.rcn.net (207.172.15.50) 17.606 ms
   bdle6.core2.phdl.pa.rcn.net (207.172.15.35) 27.232 ms
5 hge0-3-0-1.core2.lnh.md.rcn.net (207.172.19.92) 16.296 ms 18.279 ms
   207.172.19.104 (207.172.19.104) 17.031 ms
6 hge0-4-0-0.border1.eqnx.va.rcn.net (207.172.19.88) 17.137 ms
   207.172.19.24 (207.172.19.24) 16.942 ms
   207.172.19.22 (207.172.19.22) 27.964 ms
7 eqix-dc5.automattic.com (206.126.237.124) 16.025 ms 15.469 ms 16.003 ms
8 wordpress.com (198.181.119.91) 19.745 ms 22.553 ms 21.016 ms
9 100.68.10.1 (100.68.10.1) 17.913 ms 16.035 ms
   100.68.10.3 (100.68.10.3) 17.407 ms
10 wordpress.com (192.0.79.33) 15.763 ms 17.687 ms 16.110 ms

My house to NYPost web site
Testing your networking (cheating #1)

Set your laptop network connection to be on the same ‘subnet’ as the Arduino. (e.g. 198.1.100.5 for an arduino at 198.1.100.2)

Send direct to/from arduino and laptop (using webtools and ip address of the arduino from the laptop)

Testing your networking (cheating #2)

Print to your serial port instead and we’ll all just make believe
What's this do?

```cpp
#include "LPD8806.h"
#include "SPI.h"

int nLEDs = 160;
int dataPin = 2;
int clockPin = 7;

LPD8806 strip = LPD8806(nLEDs, dataPin, clockPin);

void setup() {
  // Start up the LED strip
  strip.begin();

  // Update the strip, to start they are all 'off'
  strip.show();
}

void loop() {
  sample1();
  sample2();
}

void sample1 () {
  colorChase(strip.Color(127, 0, 0), 100); // Red
  colorChase(strip.Color( 0,127, 0), 100); // Green
  colorChase(strip.Color( 0, 0,127), 100); // Blue
  colorChase(strip.Color(127,127,127), 100); // White
  }
```
void sample2() {
    int i;

    // set all red
    for (i=0; i < strip.numPixels(); i++) strip.setPixelColor(i, 127, 0, 0);
    strip.show();
    delay(5000);

    // set all blue
    for (i=0; i < strip.numPixels(); i++) strip.setPixelColor(i, 0, 0, 127);
    strip.show();
    delay(5000);

    // set all white
    for (i=0; i < strip.numPixels(); i++) strip.setPixelColor(i, 127, 127, 127);
    strip.show();
    delay(5000);
}

void colorChase(uint32_t c, uint8_t wait) {
    int i;

    // Start by turning all pixels off:
    for (i=0; i < strip.numPixels(); i++) strip.setPixelColor(i, 0);

    // Then display one pixel at a time:
    for (i=0; i < strip.numPixels(); i++) {
        strip.setPixelColor(i, c);
        strip.show();
        strip.setPixelColor(i, 0);
        delay(wait);
    }

    strip.show();
}
//LED Pin Variables
int ledPins[] = {2,3,4,5,6,7,8,9}

void setup()
{

for(int i = 0; i < 8; i++)
   pinMode(ledPins[i],OUTPUT)

}

void loop() // run over and over again
{

oneAfterAnotherLoop()

{ delay(1000);

   oneOnAtATime()
   delay(1000);

   inAndOut();
   inAndOut();
   inAndOut();
   inAndOut();
   delay(1000);

Serial.println("countup starts");
   countup();
}
void oneAfterAnotherNoLoop(){
    int delayTime = 100;
    digitalWrite(ledPins[0], HIGH))
    delay(delayTime); //waits delayTime milliseconds
    digitalWrite(ledPins[1], HIGH);
    delay(delayTime); //waits delayTime milliseconds
    digitalWrite(ledPins[2], HIGH)
    delay(delayTime); //waits delayTime milliseconds
    digitalWrite(ledPins[3], HIGH);
    delay(delayTime); //waits delayTime milliseconds
    digitalWrite(ledPins[4], HIGH);
    delay(delayTime); //waits delayTime milliseconds
    digitalWrite(ledPins[5], HIGH);
    delay(delayTime); //waits delayTime milliseconds
    digitalWrite(ledPins[6], HIGH);
    delay(delayTime); //waits delayTime milliseconds
    digitalWrite(ledPins[7], HIGH);
    delay(delayTime); //waits delayTime milliseconds

    //Turns Each LED Off
    digitalWrite(ledPins[7], LOW);
    delay(delayTime); //waits delayTime milliseconds
    digitalWrite(ledPins[6], LOW);
    delay(delayTime); //waits delayTime milliseconds
    digitalWrite(ledPins[5], LOW)
    delay(delayTime); //waits delayTime milliseconds
    digitalWrite(ledPins[4], LOW);
    delay(delayTime); //waits delayTime milliseconds
    digitalWrite(ledPins[3], LOW);
    delay(delayTime); //waits delayTime milliseconds
    digitalWrite(ledPins[2], LOW);
    delay(delayTime); //waits delayTime milliseconds
    digitalWrite(ledPins[1], LOW);
    delay(delayTime); //waits delayTime milliseconds
    digitalWrite(ledPins[0], LOW);
    delay(delayTime); //waits delayTime milliseconds
}
}
void oneAfterAnotherLoop(){
    int delayTime = 100; //the time (in milliseconds) to pause between LEDs
    //make smaller for quicker switching and larger for slower

    //Turn Each LED on one after another
    for(int i = 0; i <= 7; i++){
        digitalWrite(ledPins[i], HIGH);
        delay(delayTime);
    }

    for(int i = 7; i >= 0; i--)
        digitalWrite(ledPins[i], LOW);
    delay(delayTime);
}

void oneOnAtATime(){
    int delayTime = 100; //the time (in milliseconds) to pause between LEDs
    //make smaller for quicker switching and larger for slower

    for(int i = 0; i <= 7; i++){
        int offLED = i - 1;
        if(i == 0) {         //for i = 1 to 7 this is i minus 1 (i.e. if i = 2 we will
            offLED = 7;
        }

        digitalWrite(ledPins[i], HIGH);
        digitalWrite(ledPins[offLED], LOW);
        delay(delayTime);
    }
}
void inAndOut()
{
    int delayTime = 100;

    for(int i = 0; i <= 3; i++)
    {
        int offLED = i - 1;
        if(i == 0) {
            offLED = 3;
        }

        int onLED1 = 3 - i;
        int onLED2 = 4 + i
        int offLED1 = 3 - offLED;
        int offLED2 = 4 + offLED;

        digitalWrite(ledPins[onLED1], HIGH);
        digitalWrite(ledPins[onLED2], HIGH);
        digitalWrite(ledPins[offLED1], LOW);
        digitalWrite(ledPins[offLED2], LOW);
        delay(delayTime);
    }
}
for(int i = 3; i >= 0; i--){
    int offLED = i + 1;

    if(i == 3) {
        offLED = 0;
    }

    int onLED1 = 3 - i;
    int onLED2 = 4 + i;

    int offLED1 = 3 - offLED;
    int offLED2 = 4 + offLED;

    digitalWrite(ledPins[onLED1], HIGH);
    digitalWrite(ledPins[onLED2], HIGH);
    digitalWrite(ledPins[offLED1], LOW);
    digitalWrite(ledPins[offLED2], LOW);
    delay(delayTime);
}

void countup()
{
    int i;

    for (int ii=0; ii<255; ii++)
    { whatpins(ii);
      delay(100);
    }
void whatpins(int iii)
{
    if ((iii / 128) == 1) { digitalWrite(ledPins[7], HIGH);
        iii = iii - 128; }
    else digitalWrite(ledPins[7], LOW);

    if ((iii / 64) == 1) { digitalWrite(ledPins[6], HIGH);
        iii = iii - 64; }
    else digitalWrite(ledPins[6], LOW);

    if ((iii / 32) == 1) { digitalWrite(ledPins[5], HIGH);
        iii = iii - 32; }
    else digitalWrite(ledPins[5], LOW);

    if ((iii / 16) == 1) { digitalWrite(ledPins[4], HIGH);
        iii = iii - 16; }
    else digitalWrite(ledPins[4], LOW);

    if ((iii / 8) == 1) { digitalWrite(ledPins[3], HIGH);
        iii = iii - 8; }
    else digitalWrite(ledPins[3], LOW);

    if ((iii / 4) == 1) { digitalWrite(ledPins[2], HIGH);
        iii = iii - 4; }
    else digitalWrite(ledPins[2], LOW);

    if ((iii / 2) == 1) { digitalWrite(ledPins[1], HIGH);
        iii = iii - 2; }
    else digitalWrite(ledPins[1], LOW);

    if (iii == 1) { digitalWrite(ledPins[0], HIGH);
        iii = iii - 1; }
    else digitalWrite(ledPins[0], LOW); }

And this one?

```
/*
   Arduino thermistor example software
   Copyright (c) 2010 Mark McComb, hacktronics LLC
   License: http://www.opensource.org/licenses/mit-license.php (Go crazy)

   includes code from arduino.cc tutorial on a web server
*/

#include <SPI.h>
#include <Ethernet.h>
const int trigPin = 2;
const int echoPin = 4;

// Enter a MAC address and IP address for your controller below.
// The IP address will be dependent on your local network:
byte mac[] = { 0x90, 0xAD, 0xDA, 0x0D, 0x10, 0xAC };
IPAddress ip(192, 168, 2, 5);
long distance = 0;

// Initialize the Ethernet server library
// with the IP address and port you want to use
// (port 80 is default for HTTP):
EthernetServer server(80);

void setup(void) {
    Serial.begin(9600);

    // start the Ethernet connection and the server:
    Ethernet.begin(mac, ip);
    server.begin();
    Serial.print("server is at ");
    Serial.println(Ethernet.localIP());
}
```
long microsecondsToCentimeters(long microseconds) {
    return microseconds / 29 / 2;
}

long getDistance() {
    long duration;

    // The sensor is triggered by a HIGH pulse of 10 or more microseconds.
    // Give a short LOW pulse beforehand to ensure a clean HIGH pulse:
    pinMode(trigPin, OUTPUT);
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    // Read the signal from the sensor: a HIGH pulse whose
    // duration is the time (in microseconds) from the sending
    // of the ping to the reception of its echo off of an object.
    pinMode(echoPin, INPUT);
    duration = pulseIn(echoPin, HIGH);

    // convert the time into a distance
    Serial.println(microsecondsToCentimeters(duration));
    return microsecondsToCentimeters(duration);
}
void loop(void) {

  // listen for incoming clients
  EthernetClient client = server.available();
  if (client) {
    Serial.println("new client");
    // an http request ends with a blank line

      // send a standard http response header
      client.println("HTTP/1.1 200 OK");
      client.println("Content-Type: text/html");
      client.println("Connection: close");
      client.println();
      
      /*
      client.println("<!DOCTYPE HTML>
      client.println("<html>
      // add a meta refresh tag, so the browser pulls again every 10 seconds:
      client.println("<meta http-equiv="refresh" content="10">")
      */
      client.print("Distance: ");
      client.print(getDistance());
      
      client.println(" cm");
      /*
      client.println("<br />
      */
  
    // give the web browser time to receive the data
    delay(1);
  // close the connection:
  client.stop();
  Serial.println("client disconnected");
  }
}
#!/bin/csh

# to get distance and put it in /Users/charlermcgrew/Desktop/Dist.txt
#

while (1)

    wget -T 1 http://192.168.2.5
grep Distance index.html
if ($status == 0) then  # found "Distance"
    awk '{ print $2 }' < index.html > /Users/mcgrew/Desktop/Dist.txt
else
    echo "Can't find distance!"
endif

    rm -f index.html
    sleep 1

end

echo "Done"
What does the user-seen screen need?

- Current **State**
- Current distance
- Current State’s video
- Conditions to go to another state