```
// Include the libraries we need
#include <OneWire.h>
#include <DallasTemperature.h>
#include <Servo.h>

Servo myservo;  // create servo object to control a servo
#define servoPin 3 //~

const int minAngle = 20; const int maxAngle = 160; const int GREEN = 11; const int
YELLOW = 10; const int RED = 9; const int blinkInterval = 500; const int blinkDuration =
500; // number of millisecs that Led's are on - all three leds use this

byte RED_State = LOW;        //   LOW = off byte YELLOW_State = LOW;
byte GREEN_State = LOW;

unsigned long currentMillis = 0;          // stores the value of millis() in each iteration of loop()
unsigned long TotalRedMillis = 0; unsigned long TotalYellowMillis = 0; unsigned long
TotalGreenMillis = 0;
int buttonPushed =0; int ServoMaxDelay = 1800; int temperaturePushed =
0; int pushButtonPin = 2; int angle = 20;        // initial angle  for servo int
OnlyOnce = 0;

// Data wire is plugged into port 4 on the Arduino
#define ONE_WIRE_BUS 4

// Setup a oneWire instance to communicate with any
OneWire devices (not just Maxim/Dallas temperature ICs)
OneWire oneWire(ONE_WIRE_BUS);

// Pass our oneWire reference to Dallas Temperature.
DallasTemperature sensors(&oneWire);

// arrays to hold device address DeviceAddress insideThermometer;
```

```
/*
 * Setup function. Here we do the basics
 */
void
setup(
void)
{
  // start serial port
  Serial.begin(9600);
  Serial.println("Dallas Temperature IC Control Library Demo");


  // locate devices on the bus
  Serial.print("Locating devices...");   sensors.begin();
Serial.print("Found ");
  Serial.print(sensors.getDeviceCount(), DEC);
  Serial.println(" devices.");


  // report parasite power requirements   Serial.print("Parasite power
is: ");   if (sensors.isParasitePowerMode()) Serial.
println("ON");   else
Serial.println("OFF");


  // Assign address manually. The addresses below will beed to be changed
  // to valid device addresses on your bus. Device address can be retrieved
  // by using either oneWire.search(deviceAddress) or individually via
  // sensors.getAddress(deviceAddress, index)   // Note that you will need to use
your specific address here
  //insideThermometer = { 0x28, 0x1D, 0x39, 0x31, 0x2,
0x0, 0x0, 0xF0 };


  // Method 1:
  // Search for devices on the bus and assign based on an index. Ideally,
  // you would do this to initially discover addresses on the bus and then
  // use those addresses and manually assign them (see above) once you know
  // the devices on your bus (and assuming they don't change).   if
(!sensors.getAddress(insideThermometer, 0)) Serial. println("Unable to find address for Device
0");
```

```
  // method 2: search()
  // search() looks for the next device. Returns 1 if a new address has been
  // returned. A zero might mean that the bus is shorted, there are no devices,
  // or you have already retrieved all of them. It might be a good idea to
  // check the CRC to make sure you didn't get garbage.
The order is
  // deterministic. You will always get the same devices in the same order
  //
  // Must be called before search()
  //oneWire.reset_search();
  // assigns the first address found to insideThermometer
  //if (!oneWire.search(insideThermometer)) Serial. println("Unable to find address for
insideThermometer");


  // show the addresses we found on the bus   Serial.print("Device 0
Address: ");   printAddress(insideThermometer);   Serial.println();


  // set the resolution to 9 bit (Each Dallas/Maxim device is capable of several different
resolutions)   sensors.setResolution(insideThermometer, 9);


  Serial.print("Device 0 Resolution: ");   Serial.print(sensors.
getResolution(insideThermometer), DEC);   Serial.println();




  // Servo button demo
  Serial.begin(9600);          //  setup serial   myservo.attach(servoPin);  // attaches the
servo on pin 3 to the servo object   pinMode(pushButtonPin,INPUT_PULLUP);
pinMode(RED, OUTPUT);   pinMode(YELLOW, OUTPUT);   pinMode(GREEN, OUTPUT);
myservo.write(angle);




}
// function to print the temperature for a device void printTemperature(DeviceAddress
deviceAddress)
{
  // method 1 - slower
```

```
  //Serial.print("Temp C: ");
  //Serial.print(sensors.getTempC(deviceAddress));
  //Serial.print(" Temp F: ");
  //Serial.print(sensors.getTempF(deviceAddress)); //
Makes a second call to getTempC and then converts to
Fahrenheit
  // method 2 - faster   float tempC = sensors.getTempC(deviceAddress);
if(tempC == DEVICE_DISCONNECTED_C)
  {
          Serial.println("Error: Could not read temperature data");      return;
  }
  Serial.print("Temp C: ");
  Serial.println(tempC);



//=====================================================
============================================================
=====================================================
  {   if(digitalRead(pushButtonPin) == LOW){
buttonPushed = 1;
  }   if(
buttonPushed ){
  // change the angle for next time through the loop:   for (angle = 20; angle <= 160; angle += 1)
{ // goes from 0 degrees to 180 degrees        // in steps of 1 degree
myservo.write(angle);             // tell servo to go to position in variable 'pos'     delay(15);
          // waits 15ms for the servo to reach the position
   }    if (angle =
160){    delay
(10000);
   if (currentMillis - ServoMaxDelay >= 0){     for (angle = 150; angle >= 20;
angle -= 1) {    myservo.write(angle);                        delay(15);
buttonPushed = 0;


          }
          }
  }
   }
{ if (OnlyOnce == 0) {  if ((tempC >= 30) &&
(tempC<=40)){  temperaturePushed = 1;
```

```
 }


   if( temperaturePushed ){
  // change the angle for next time through the loop:   for (angle = 20; angle <= 160; angle += 1)
{ // goes from 0 degrees to 180 degrees     // in steps of 1 degree     myservo.write(angle);
// tell servo to go to position in variable 'pos' delay(15);                  // waits 15ms for the servo
to reach the position
   }    if (angle = 160){  delay (10000); if (currentMillis - ServoMaxDelay >= 0){
          for (angle = 150; angle >= 20; angle -= 1) {     myservo.write(angle);
delay(15);
    buttonPushed = 0;  OnlyOnce++;
          }
          }
  }
   }
   }
}


}




{ //RED
LED   if
(tempC<2
4.5){


if (RED_State == LOW) {       if (currentMillis - TotalRedMillis
>= blinkInterval) {       RED_State = HIGH;
digitalWrite (RED, HIGH);       TotalRedMillis += blinkInterval;
          }   }   else {       if (currentMillis - TotalRedMillis >=
blinkDuration) {          RED_State = LOW;    digitalWrite (RED,
LOW);             digitalWrite (GREEN, LOW);
digitalWrite (YELLOW, LOW);           TotalRedMillis +=
blinkDuration;
          }
  }
  }
}
```

```
//YEL
LOW
LED if
(temp
C>35.
5){


if (YELLOW_State == LOW) {          if (currentMillis -
TotalYellowMillis >= blinkInterval) {
        YELLOW_State = HIGH;       digitalWrite (YELLOW,
HIGH);          TotalYellowMillis += blinkInterval;
        }   }   else {      if (currentMillis - TotalYellowMillis >=
blinkDuration) {
        YELLOW_State = LOW;        digitalWrite
(YELLOW, LOW);       digitalWrite (GREEN, LOW);
        digitalWrite (RED, LOW);
    TotalYellowMillis += blinkDuration;
        }
 }
 }



//GREEN LED if (tempC>=25 &&
tempC<=35){ if (GREEN_State
== LOW) {     if (currentMillis -
TotalGreenMillis >= blinkInterval)
{
        GREEN_State = HIGH;        digitalWrite (GREEN,
HIGH);          TotalGreenMillis += blinkInterval;
        }   }   else {      if (currentMillis - TotalGreenMillis >=
blinkDuration) {
        GREEN_State = LOW;         digitalWrite
(GREEN, LOW);        digitalWrite (YELLOW, LOW);
        digitalWrite (RED, LOW);
    TotalGreenMillis += blinkDuration;
        }
 }
```

```
}

}



void loop(){ currentMillis =
millis();
{
  // call sensors.requestTemperatures() to issue a global temperature
  // request to all devices on the bus   sensors.requestTemperatures(); // Send the command to
get temperatures




  // It responds almost immediately. Let's print out the data
printTemperature(insideThermometer); // Use a simple function to print out the data






}
Serial.println(OnlyOnce);
}
// function to print a device address void printAddress(DeviceAddress
deviceAddress)
{   for (uint8_t i = 0; i < 8; i++)
  {
          if (deviceAddress[i] < 16) Serial.print("0");
    Serial.print(deviceAddress[i], HEX);
  }
}
```