

GETTING STARTED WITH ARDUINO

MATERIALS

Part Name	Cost	Where to Buy
Arduino UNO	\$29.95	https://www.sparkfun.com/products/11021
USB Cable	\$3.95	https://www.sparkfun.com/products/512
LED	\$0.35	https://www.sparkfun.com/products/9590
RHT03 Temperature and Humidity Sensor	\$9.95	https://www.sparkfun.com/products/10167
1k Resistor	\$0.25	https://www.sparkfun.com/products/8374
Hookup Wire	\$6.95	https://www.sparkfun.com/products/124
Breadboard	\$3.95	https://www.sparkfun.com/products/11658
Serial LCD	\$15.95	https://www.sparkfun.com/products/709
10k Potentiometer	\$0.95	https://www.sparkfun.com/products/9806
MicroSD Arduino Shield	\$16.95	https://www.sparkfun.com/products/9899
8GB MicroSD card with USB reader	\$9.95	https://www.sparkfun.com/products/11609

Total Cost \$99.15

INSTALLING THE SOFTWARE

- We need to install the Arduino software.
- That can be done by visiting [here](#).
- Extract the zip file that you downloaded.

THE Arduino SOFTWARE IS PROVIDED TO YOU "AS IS," AND WE MAKE NO EXPRESS OR IMPLIED WARRANTIES WHATSOEVER WITH RESPECT TO ITS FUNCTIONALITY, OPERABILITY, OR USE, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR INFRINGEMENT. WE EXPRESSLY DISCLAIM ANY LIABILITY WHATSOEVER FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, INCIDENTAL OR SPECIAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST REVENUES, LOST PROFITS, LOSSES RESULTING FROM BUSINESS INTERRUPTION OR LOSS OF DATA, REGARDLESS OF THE FORM OF ACTION OR LEGAL THEORY UNDER WHICH THE LIABILITY MAY BE ASSERTED, EVEN IF ADVISED OF THE POSSIBILITY OR LIKELIHOOD OF SUCH DAMAGES.



By downloading the software from this page, you agree to the specified terms.

Download

Arduino 1.0.3 ([release notes](#)), hosted by [Google Code](#):

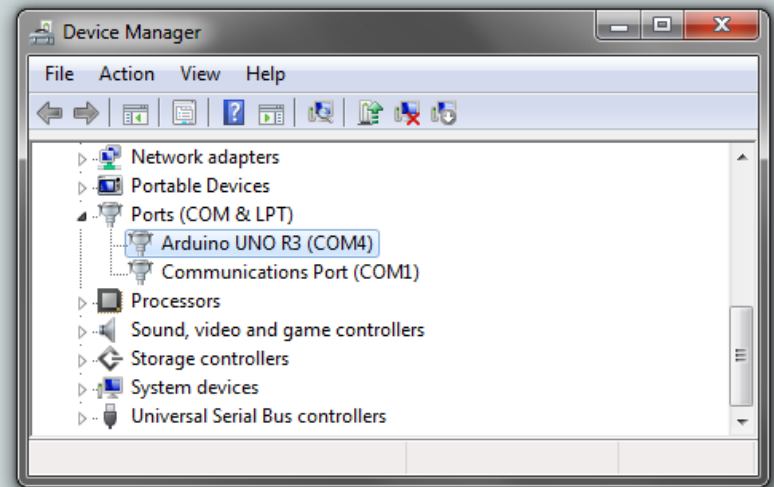
- + [Windows](#)
- + [Mac OS X](#)
- + [Linux: 32 bit, 64 bit](#)
- + [source](#)

Next steps

[Getting Started](#)
[Reference](#)
[Environment](#)
[Examples](#)
[Foundations](#)
[FAQ](#)

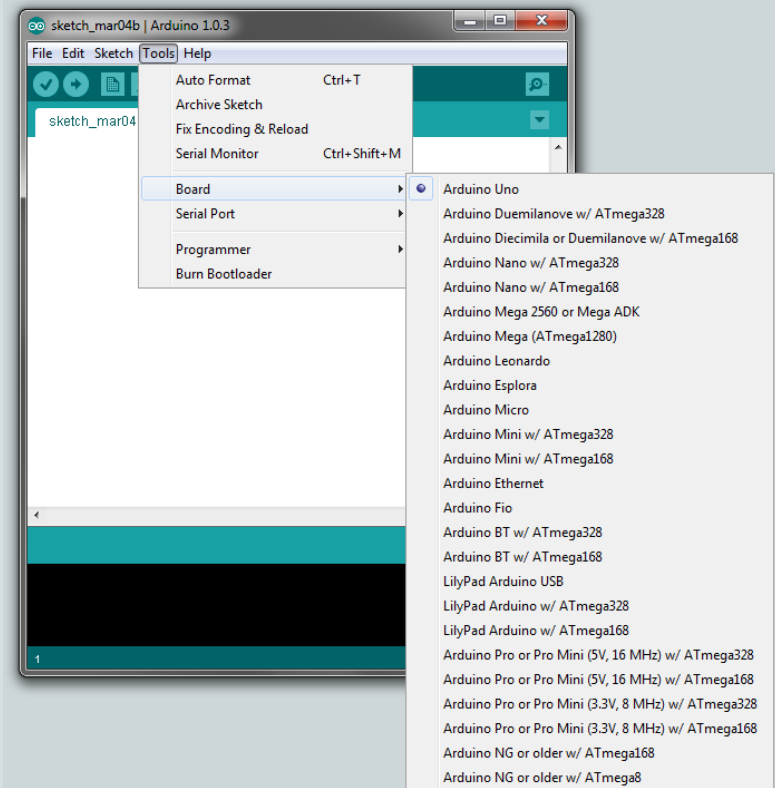
INSTALLING THE DRIVERS

- Plug in the Arduino Uno
- If using Windows, you may need to follow these directions to install the driver



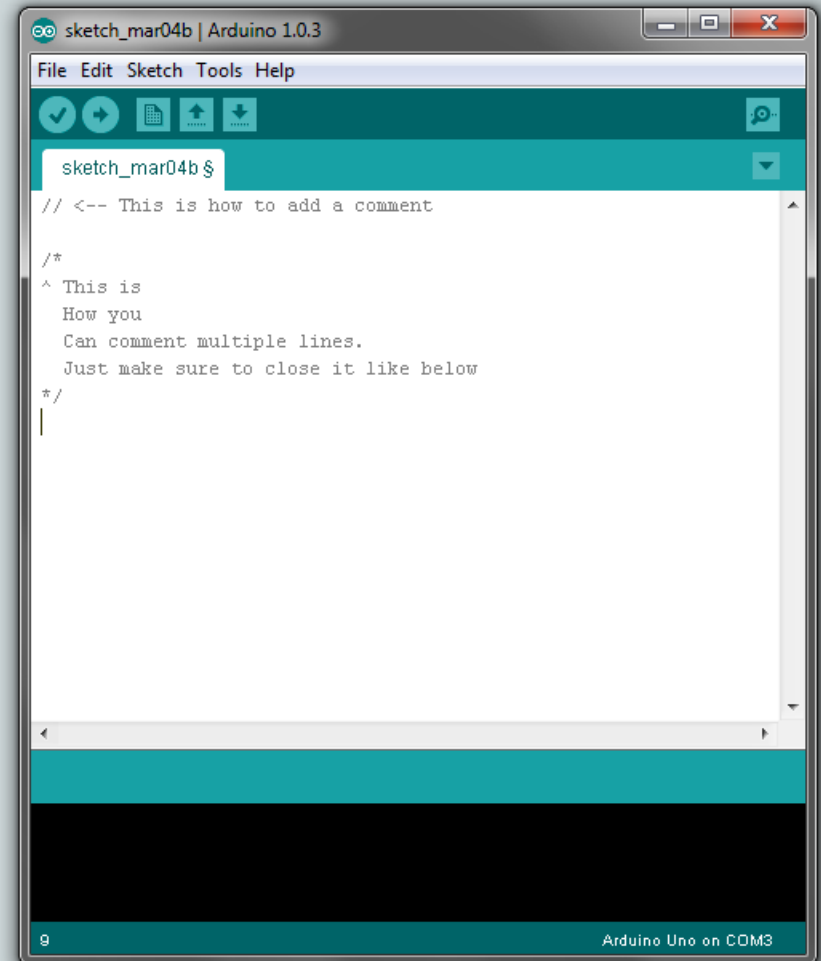
CREATING A SKETCH

- Open up the Arduino software and create a new sketch
- Make sure that under tools Board is set to your Arduino (Uno in this case)
- You may also need to change the Serial Port to your Arduino in Tools-Serial Port



COMMENTS

- Comments are very useful to remind yourself why you wrote certain pieces of code
- You can precede a line with `//` to comment the line
- You can comment an entire block of code with a `/*` and `*/`. Everything in between will be commented.



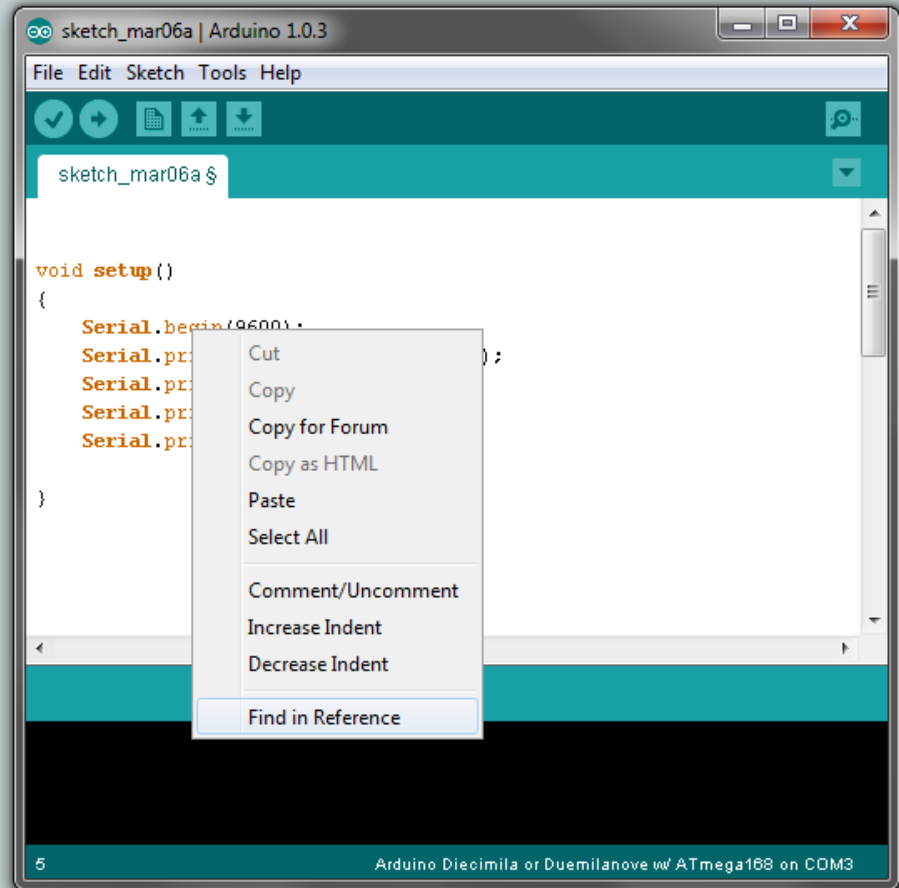
LINE END CHARACTER ;

- Moving forward you will notice a semi-colon at the end of every executable line of code
- This is mandatory and you must include the ;

```
void setup()  
{  
  Serial.begin(9600);  
  Serial.println("DHT TEST PROGRAM ");  
  Serial.print("LIBRARY VERSION: ");  
  Serial.println(DHT_LIB_VERSION);  
}
```

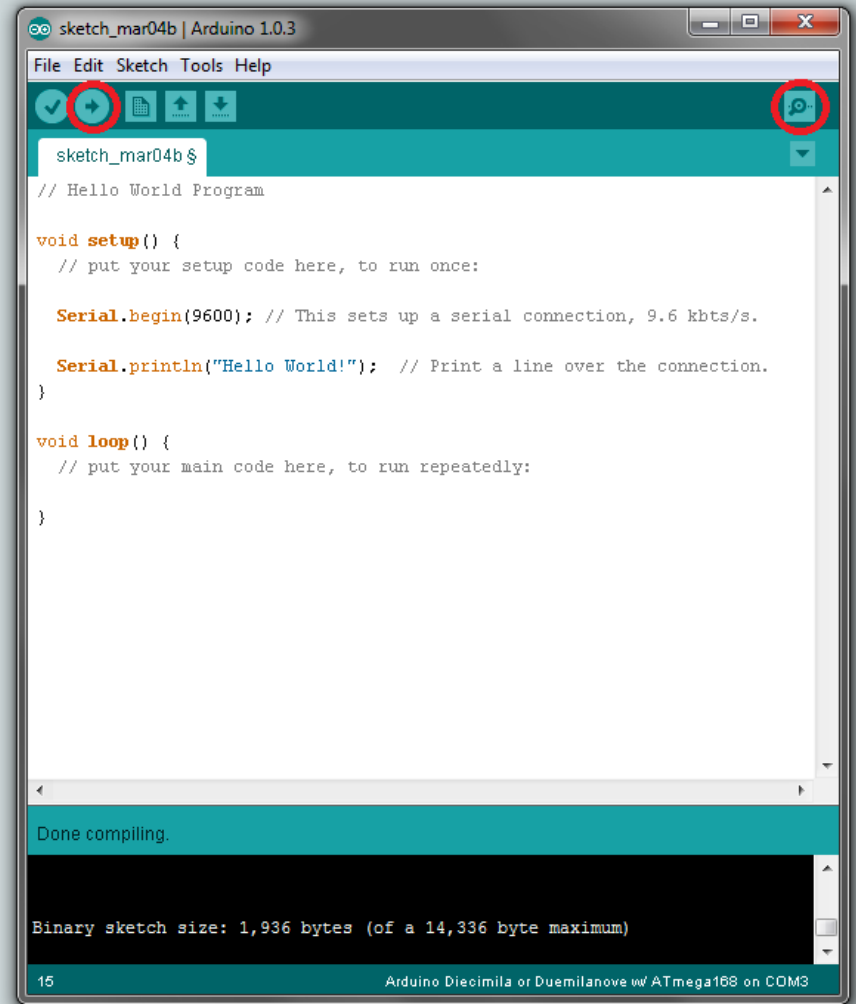
REFERENCE

- At anytime, if you do not understand a keyword, you can place your cursor on the word and right click it and select Find in Reference
- This will bring you to a website documenting the keyword and its usage



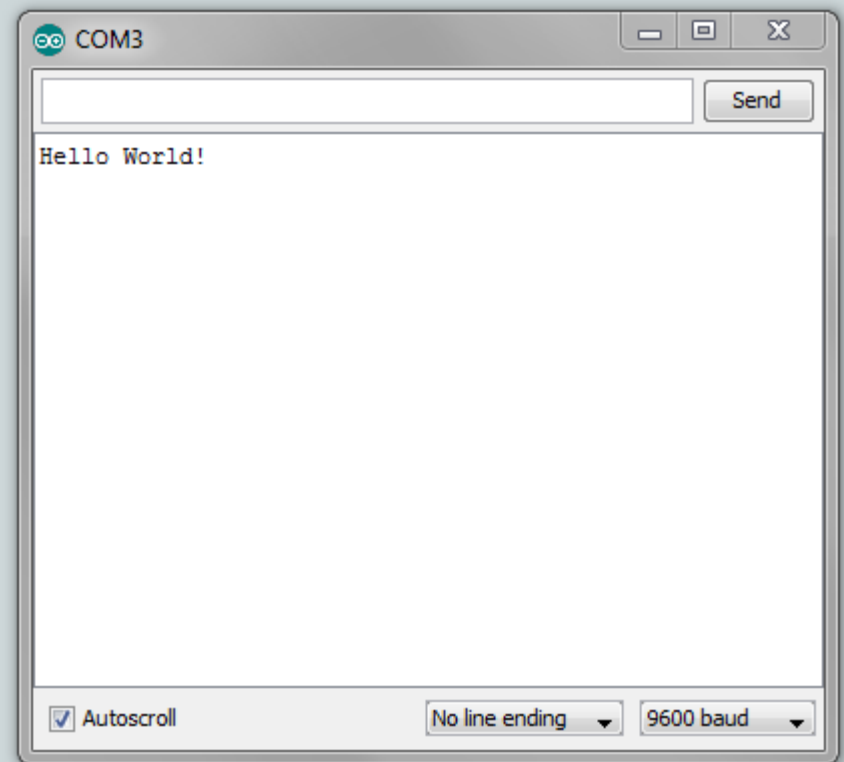
HELLO WORLD

- Enter this code into a new sketch
- Once you finish the code, you **UPLOAD** it to the Arduino with the arrow button
- You can then open the Serial monitor to view the output
- The void keyword is used to identify functions. The setup function is run first and then the loop executes over and over. This is where your main logic will reside



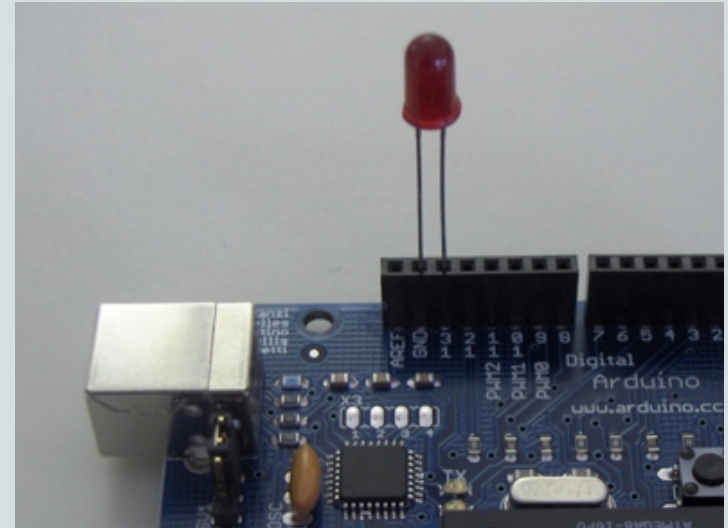
HELLO WORLD OUTPUT

- If communications are working properly, you should see this window and “Hello World”
- If you do not see the Hello World or are presented with errors, you may need to change which COM port your Arduino is on.
- You can do this under the Tools-Serial Port menu



BLINKING LED

- Plug your LED into the Arduino as shown. The LED has 2 different length leads.
- The longer one should connect to pin 13 with the shorter one in GND
- You can try reversing the leads without damaging anything



BLINKING LED (CONTINUED)

- You need to initialize the pin we are using as an OUTPUT (vs an INPUT for a sensor)
- You will then write a high value, wait 1 second, and write a low to cause the led to blink.
- When complete, you click the arrow to upload the program, once it finishes uploading it will run void setup() followed by loop()

A screenshot of the Arduino IDE interface. The title bar reads "sketch_mar04b | Arduino 1.0.3". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for opening, saving, and uploading files. The main text area contains the following code:

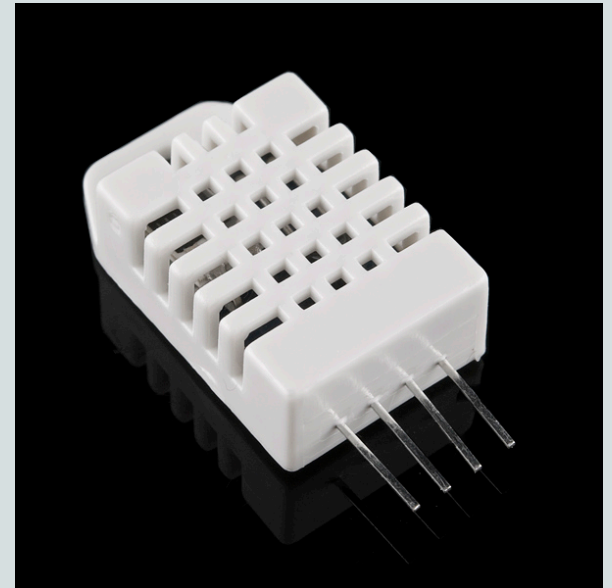
```
sketch_mar04b $  
/* Blinking LED  
 * turns on and off a light emitting diode(LED) connected to a digital  
 * pin, in intervals of 2 seconds. Ideally we use pin 13 on the Arduino  
 * board because it has a resistor attached to it, needing only an LED  
 */  
|  
int ledPin = 13;           // LED connected to digital pin 13  
  
void setup()  
{  
  pinMode(ledPin, OUTPUT); // sets the digital pin as output  
}  
  
void loop()  
{  
  digitalWrite(ledPin, HIGH); // sets the LED on  
  delay(1000);                // waits for a second  
  digitalWrite(ledPin, LOW);  // sets the LED off  
  delay(1000);                // waits for a second  
}  
  
Done compiling.  
  
Binary sketch size: 1,936 bytes (of a 14,336 byte maximum)  
  
6 Arduino Diecimila or Duemilanove w/ ATmega168 on COM3
```

BLINKING LED (CONTINUED)

Now you should have a blinking LED!

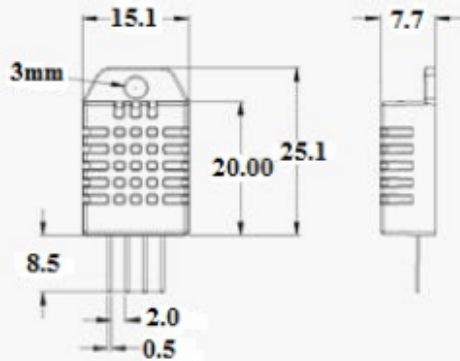
TEMPERATURE SENSOR

- You will need:
- Arduino UNO
- RHT03 Temp/Humidity Sensor
- 1k resistor
- Hookup wire
- Breadboard



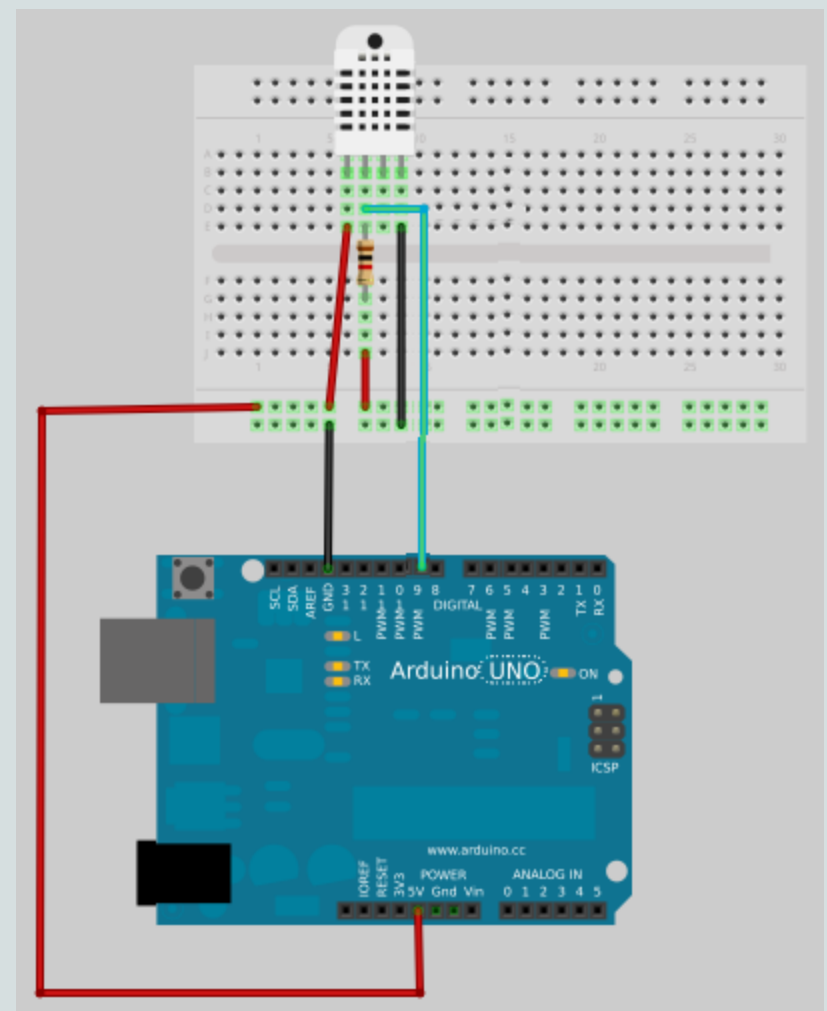
TEMPERATURE SENSOR (CONTINUED)

- First you need to wire up the RHT03 sensor as shown to the right.
- You will use pin 9 on the Arduino



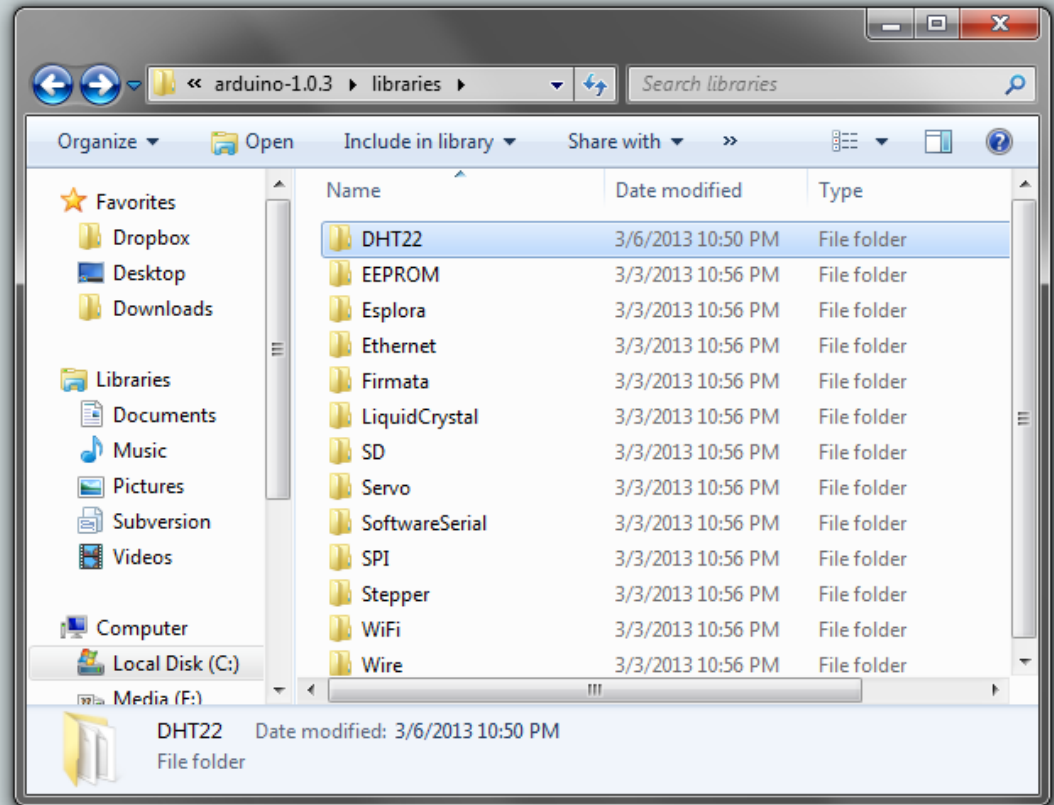
Pin sequence number: 1 2 3 4 (from left to right direction).

Pin	Function
1	VDD—power supply
2	DATA—signal
3	NULL
4	GND



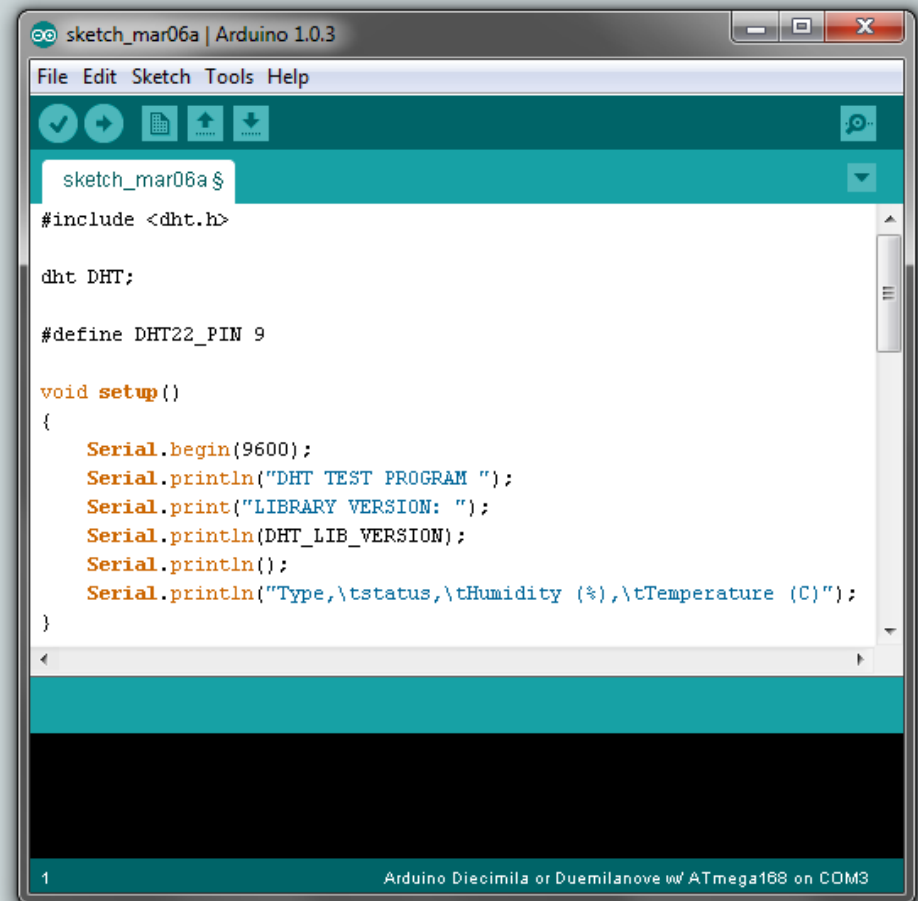
TEMPERATURE SENSOR (CONTINUED)

- To use the temperature sensor you need to copy the DHT22 folder into the libraries folder in the Arduino folder
- You should see other libraries that we will use later already copied here



TEMPERATURE SENSOR (CONTINUED)

- Once you have copied the DHT22 folder properly you need to include the library in the program
- This is done by including the appropriate header file with `#include <dht.h>`
- This file is a library that contains prewritten functions for interfacing with the sensor
- You can then write the setup function as shown



```
sketch_mar06a | Arduino 1.0.3
File Edit Sketch Tools Help
sketch_mar06a $
#include <dht.h>

dht DHT;

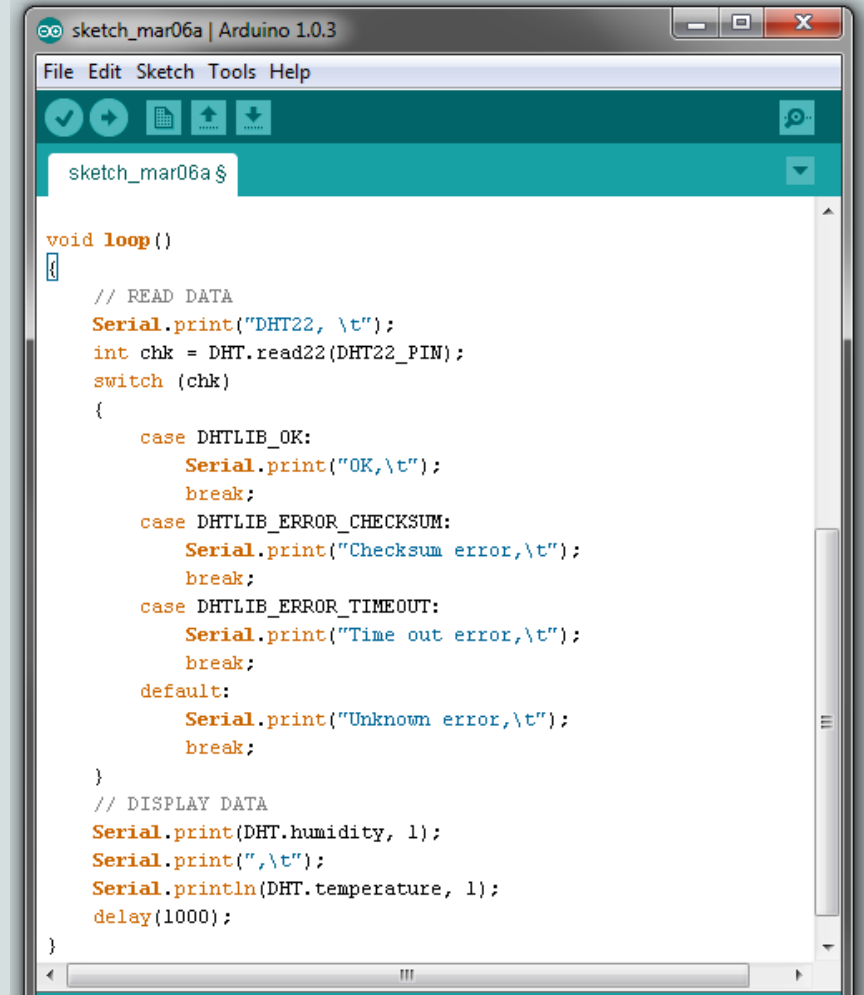
#define DHT22_PIN 9

void setup()
{
  Serial.begin(9600);
  Serial.println("DHT TEST PROGRAM ");
  Serial.print("LIBRARY VERSION: ");
  Serial.println(DHT_LIB_VERSION);
  Serial.println();
  Serial.println("Type,\tstatus,\tHumidity (%),\tTemperature (C)");
}

1 Arduino Diecimila or Duemilanove w/ ATmega168 on COM3
```

TEMPERATURE SENSOR (CONTINUED)

- The main loop of the program will call DHT.read22 on the declared pin (9 in this case)
- It will then use a switch statement to display any errors IF they occur
- It will then print the humidity and temperature data to the serial monitor which can be opened with the button in the top right

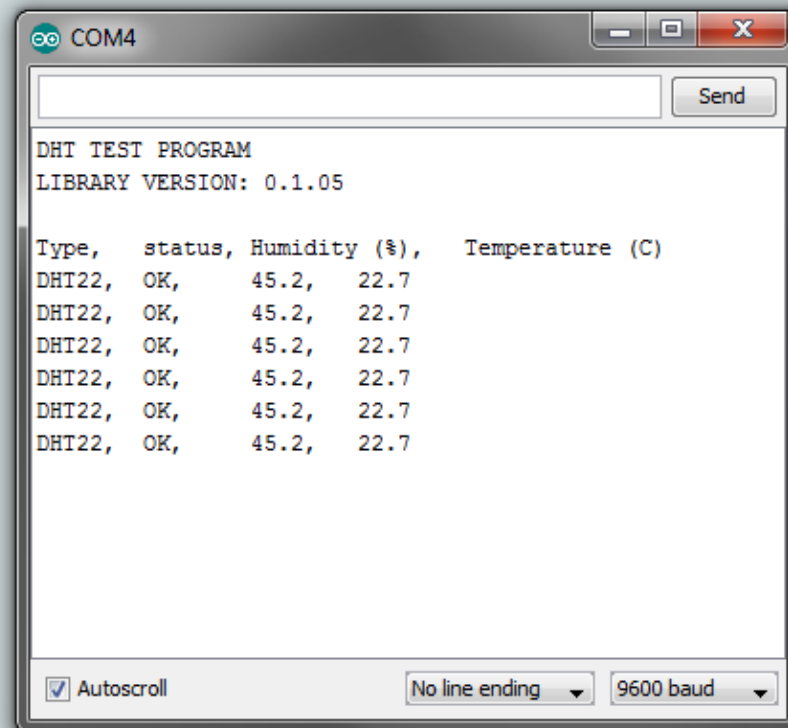


```
sketch_mar06a | Arduino 1.0.3
File Edit Sketch Tools Help
sketch_mar06a $

void loop()
{
    // READ DATA
    Serial.print("DHT22, \t");
    int chk = DHT.read22(DHT22_PIN);
    switch (chk)
    {
        case DHTLIB_OK:
            Serial.print("OK,\t");
            break;
        case DHTLIB_ERROR_CHECKSUM:
            Serial.print("Checksum error,\t");
            break;
        case DHTLIB_ERROR_TIMEOUT:
            Serial.print("Time out error,\t");
            break;
        default:
            Serial.print("Unknown error,\t");
            break;
    }
    // DISPLAY DATA
    Serial.print(DHT.humidity, 1);
    Serial.print(",\t");
    Serial.println(DHT.temperature, 1);
    delay(1000);
}
```

TEMPERATURE SENSOR (CONTINUED)

If everything goes well, you should see an output similar to this in the serial monitor window

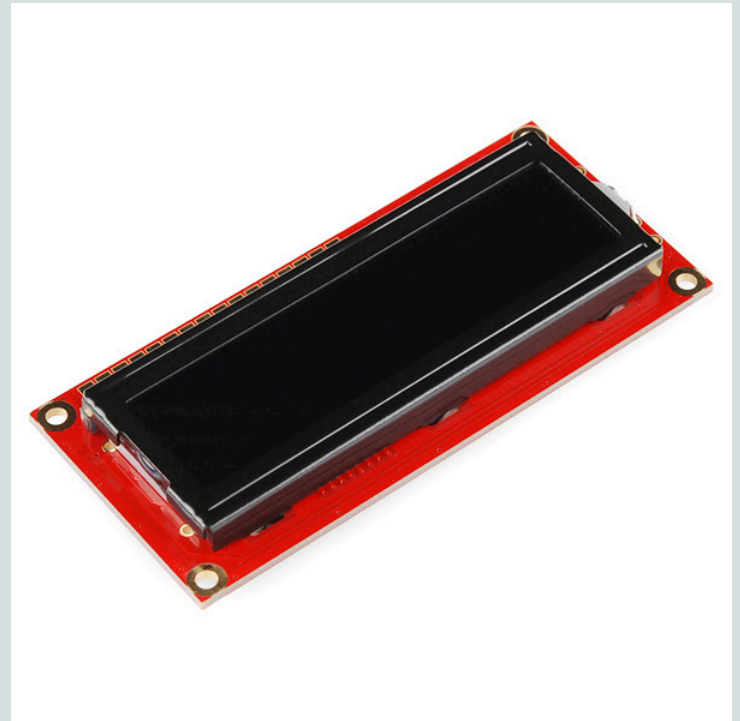


TEMPERATURE SENSOR (CONTINUED)

Leave the temperature sensor hooked up when continuing onto the next project, we will be using it later!

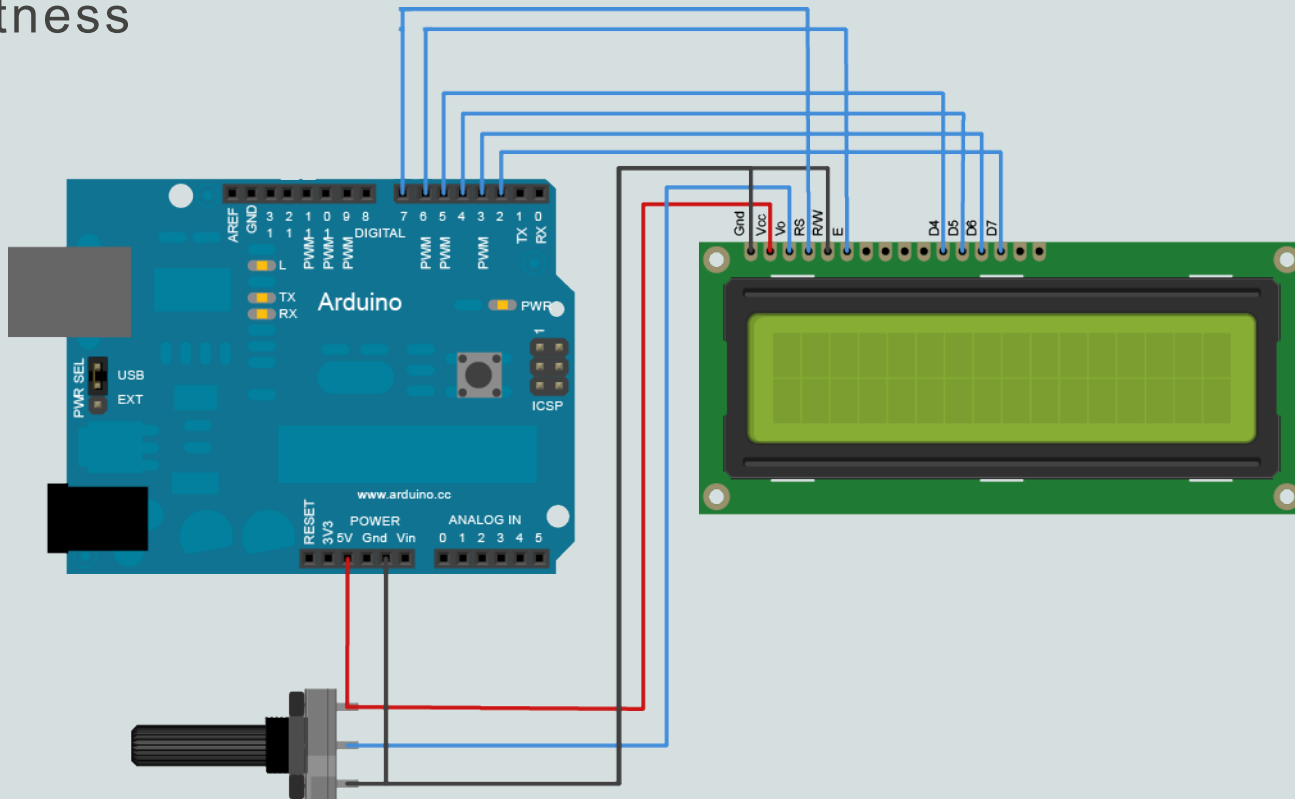
SERIAL LCD

- Now we are going to create a program that prints information to the LCD screen
- You will need:
 - Arduino UNO
 - White on black Serial LCD
 - 10k potentiometer
 - Hookup wire
 - Breadboard



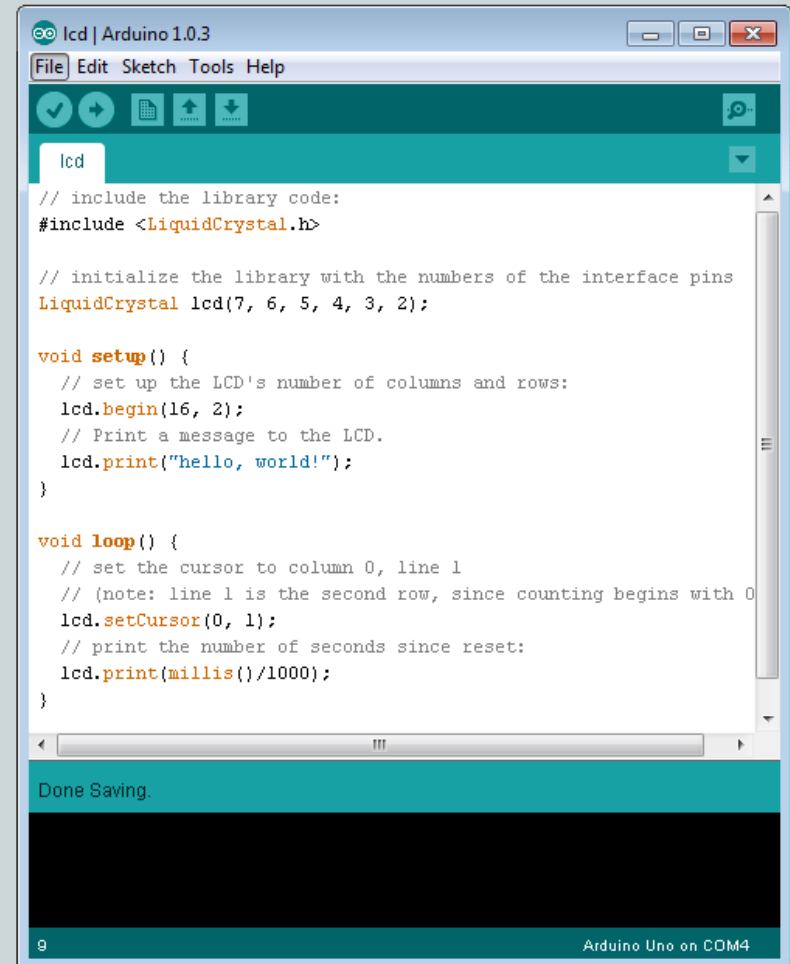
SERIAL LCD (CONTINUED)

- Create the circuit shown below using the breadboard and a soldering iron with the hookup wire
- The 10k potentiometer shown at the bottom is used to control brightness



SERIAL LCD (CONTINUED)

- Create a new sketch and save it as SerialLCD
- We will be using the LiquidCrystal.h library which is already installed so you do not need to manually drag it into the libraries folder like you need to do for the temperature sensor



The screenshot shows the Arduino IDE interface with a sketch named 'lcd'. The code is as follows:

```
// include the library code:
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(7, 6, 5, 4, 3, 2);

void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.print("hello, world!");
}

void loop() {
  // set the cursor to column 0, line 1
  // (note: line 1 is the second row, since counting begins with 0)
  lcd.setCursor(0, 1);
  // print the number of seconds since reset:
  lcd.print(millis()/1000);
}
```

At the bottom of the IDE, a status bar shows 'Done Saving.' and 'Arduino Uno on COM4'.

SERIAL LCD (CONTINUED)

- If everything works you should see something similar to this on the LCD!
- If you are experiencing issues double check all of your wiring to the breadboard, and try adjusting the potentiometer with a small screwdriver (controls the brightness)



TEMPERATURE ON SERIAL LCD

- Once both the temperature and LCD sensor are working, we are going to print the temperature and humidity data onto the LCD instead of in the serial window.
- Start with the void setup()
- Make sure to include the dht and LiquidCrystal libraries at the top!



```
lcdandtemp | Arduino 1.0.3
File Edit Sketch Tools Help

lcdandtemp $

#include <dht.h>
#include <LiquidCrystal.h>

LiquidCrystal lcd(7, 6, 5, 4, 3, 2);
dht DHT;

#define DHT22_PIN 9

void setup()
{
    lcd.begin(16, 2);
}
```

Done Saving.

18 Arduino Uno on COM4

TEMPERATURE ON SERIAL LCD (CONT)

- Now we need to create the main loop where the data is read and displayed
- Note that the conversion from Celsius to Fahrenheit need to be done since the temp sensor outputs Celsius
- Once this is done you can upload it to the Arduino and test it!



```
lcdandtemp | Arduino 1.0.3
File Edit Sketch Tools Help

void loop()
{
    // READ DATA
    int chk = DHT.read22(DHT22_PIN);
    float celsiustemp=DHT.temperature;
    float farentemp = (celsiustemp * 9.0)/ 5.0 + 32.0;

    lcd.clear();
    lcd.setCursor(0,0);

    // DISPLAY DATA
    lcd.print("Temp: ");
    lcd.print(farentemp, 1);
    lcd.print(" F");
    lcd.setCursor(0, 1);
    lcd.print("Humidity: ");
    lcd.print(DHT.humidity, 1);
    lcd.print("%");

    delay(1000);
}
```

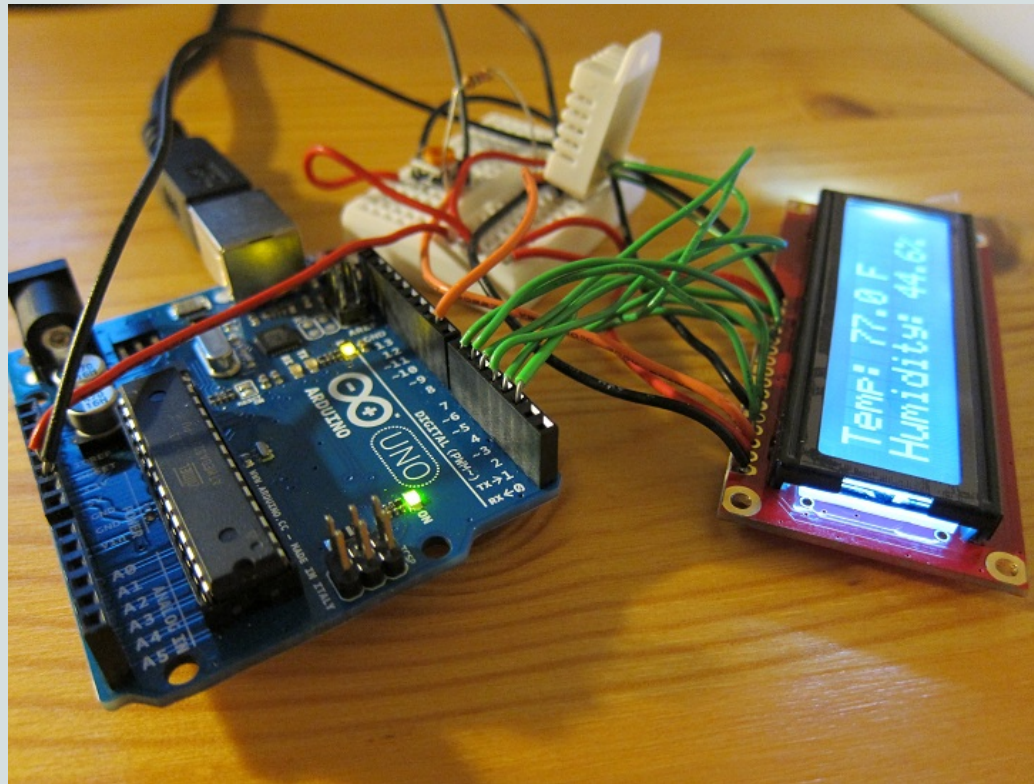
Done Saving.

Binary sketch size: 5,402 bytes (of a 32,256 byte maximum)

1 Arduino Uno on COM4

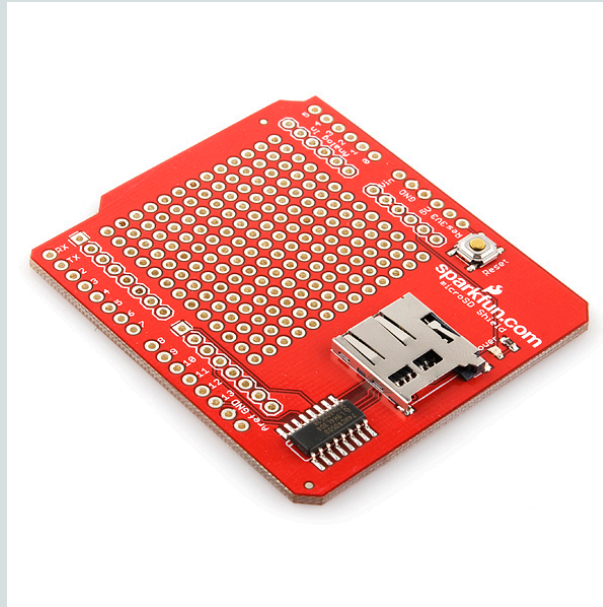
TEMPERATURE ON SERIAL LCD (CONT)

Once this is working it should look something like this...



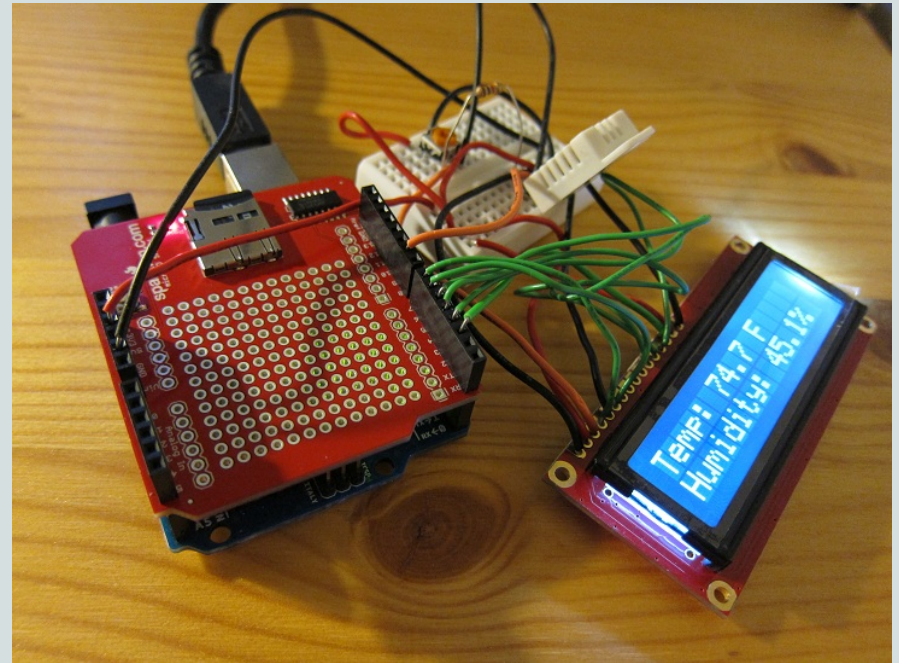
LOGGING TO μ SD CARD

Finally we are going to log the data to a micro SD card...
then you can plot it in excel or do other nerd things... I
don't know, just do it, its cool.



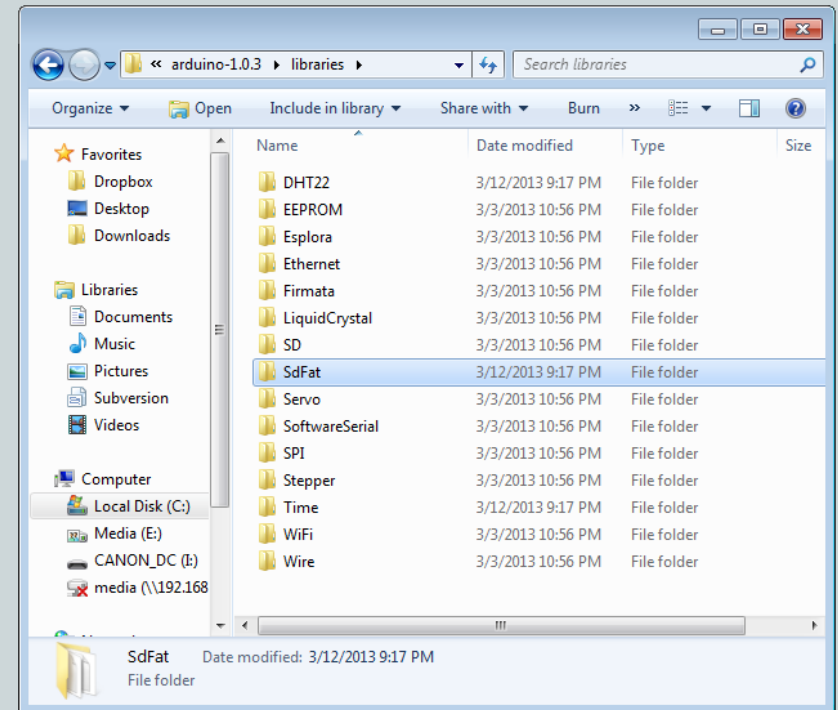
LOGGING TO μ SD CARD (CONTINUED)

- First we need to put the micro SD shield onto the Arduino.
- Once you line up the pins you need to push it onto the Arduino
- Then refer to the earlier schematic with wiring the sensor and LCD back up
- Once you plug the Arduino USB back in you should reupload your sensor/LCD sketch to make sure its still working and wired right



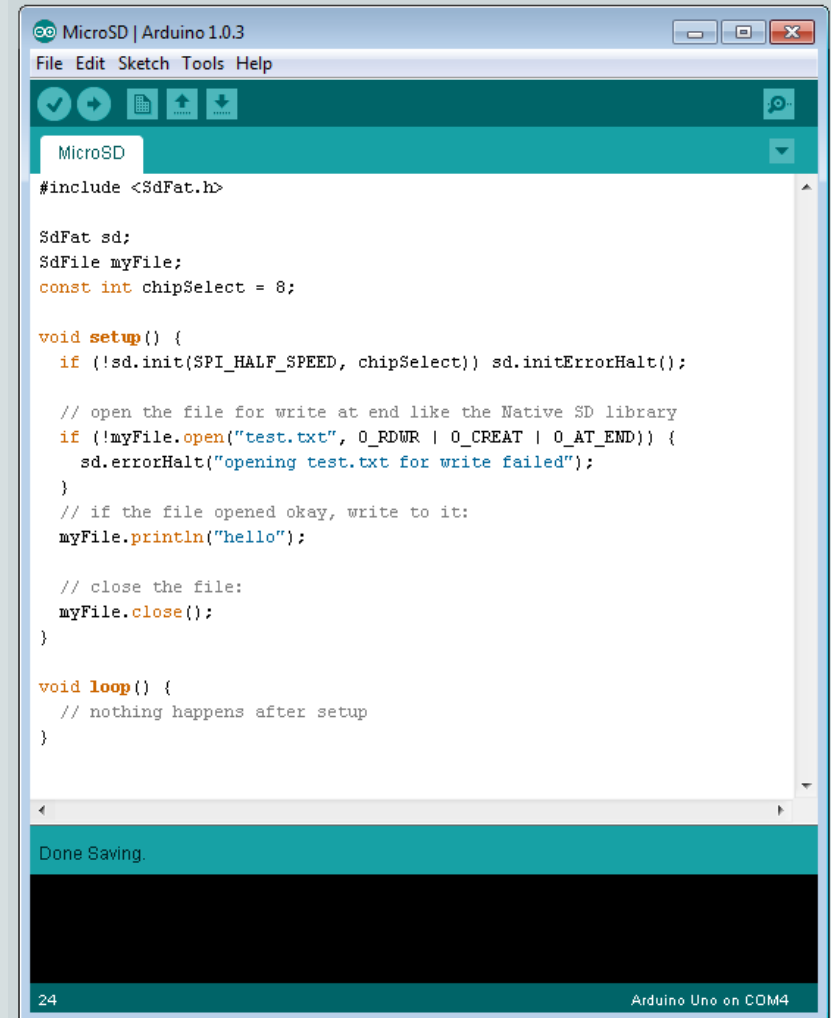
LOGGING TO μ SD CARD (CONTINUED)

- First we will just write to the SD card to test it, then we will add the code into the sensor/lcd program and create a crazy cool big program.
- Copy the “SdFat” folder into the libraries folder like we did with the DHT22 library



LOGGING TO μ SD CARD (CONTINUED)

- This code will check for a file called test.txt on the sd card.
- If it is not there, it will create it. If it is there then it will open it.
- It will then print a single line “hello” and close the file
- Nothing runs in the loop.

A screenshot of the Arduino IDE interface. The title bar reads "MicroSD | Arduino 1.0.3". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for opening, saving, and running. The main text area shows the following C++ code:

```
#include <SdFat.h>

SdFat sd;
SdFile myFile;
const int chipSelect = 8;

void setup() {
  if (!sd.init(SPI_HALF_SPEED, chipSelect)) sd.initErrorHalt();

  // open the file for write at end like the Native SD library
  if (!myFile.open("test.txt", O_RDWR | O_CREAT | O_AT_END)) {
    sd.errorHalt("opening test.txt for write failed");
  }
  // if the file opened okay, write to it:
  myFile.println("hello");

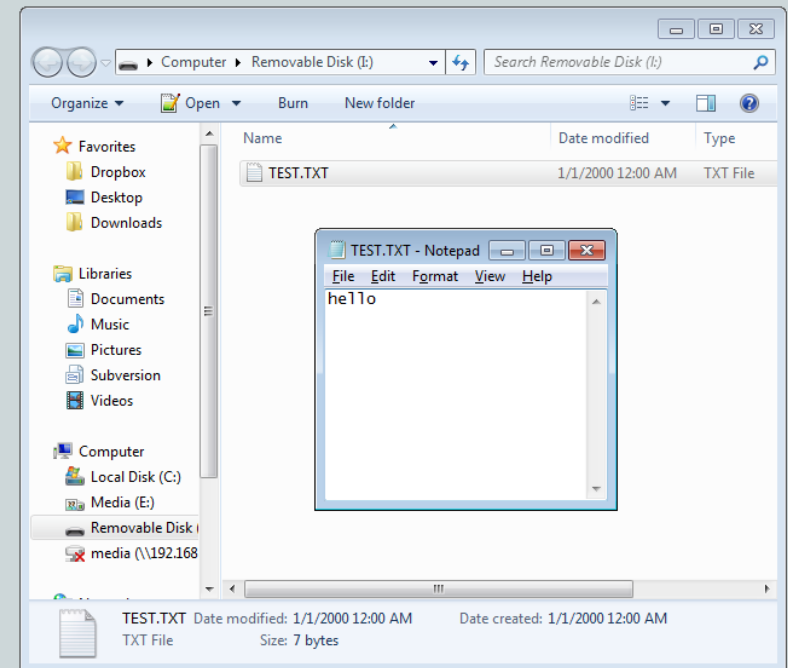
  // close the file:
  myFile.close();
}

void loop() {
  // nothing happens after setup
}
```

The status bar at the bottom shows "24" on the left and "Arduino Uno on COM4" on the right. A "Done Saving." message is visible in the bottom left of the IDE window.

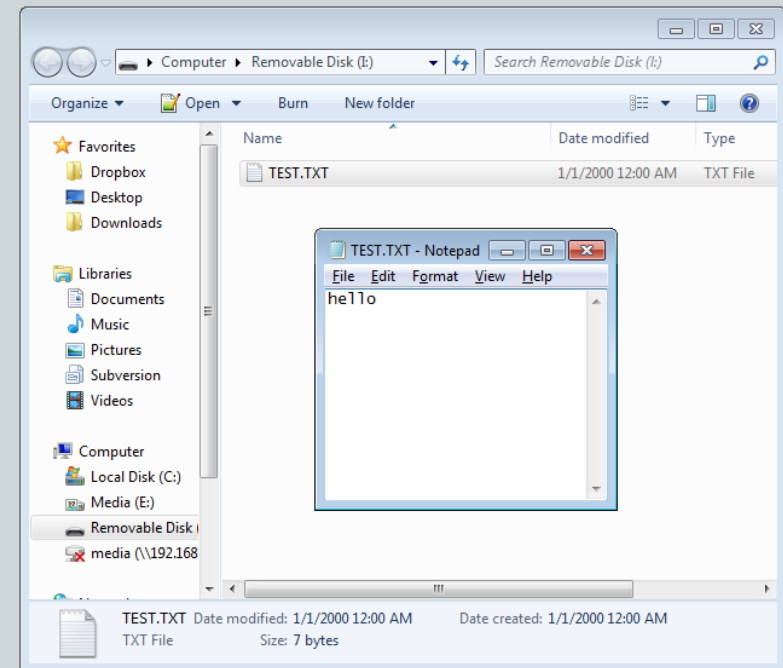
LOGGING TO μ SD CARD (CONTINUED)

- Upload this program to the Arduino
- Note: anytime you are running the Arduino with data logging like this you should probably have the SD card in the Arduino...
- Once it runs (wait until it says “Done uploading” and wait a second) you can take the μ SD card out.
- Plug it into the adapter and your computer and verify that the test file was written.



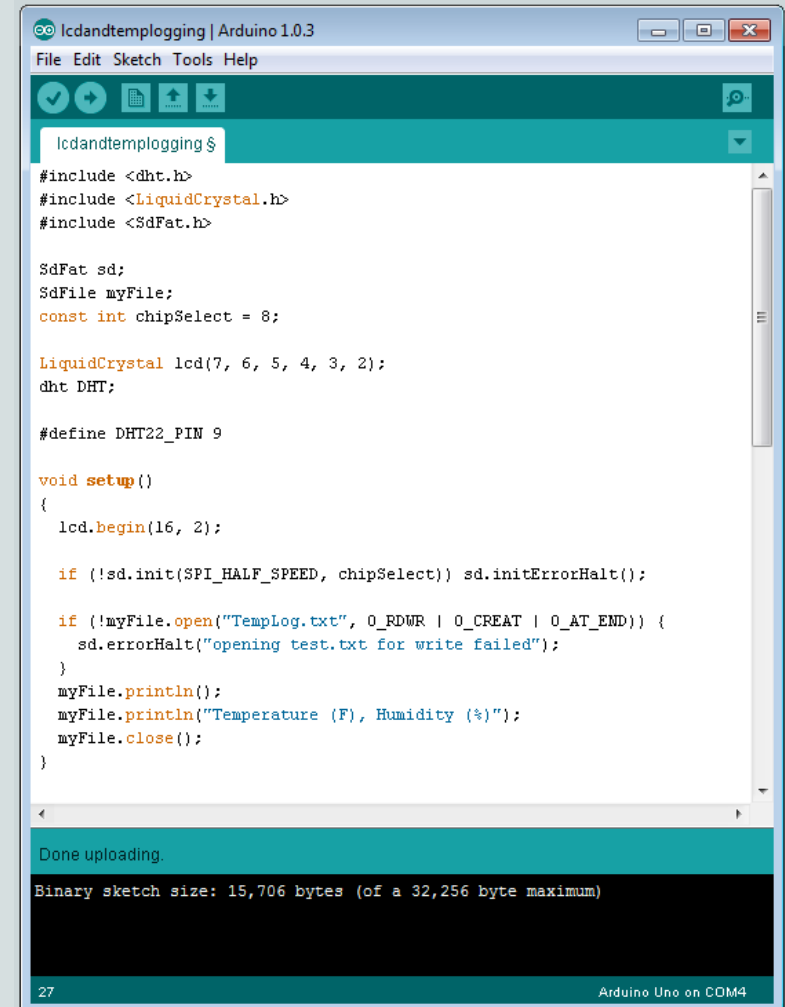
LOGGING TO μ SD CARD (CONTINUED)

- Upload this program to the Arduino
- Once it runs (wait until it says “Done uploading” and wait a second) you can take the μ SD card out.
- Plug it into the adapter and your computer and verify that the test file was written.



LOGGING TO μ SD CARD (CONTINUED)

- Next we are going to incorporate this logging to an SD card into our temperature program that displays to the lcd.
- Open the existing sketch you had made with the temperature and LCD and do a Save As so that you have a new copy with data logging.
- Begin entering the program void setup and library initialization



```
lcdandtemplogging | Arduino 1.0.3
File Edit Sketch Tools Help

#include <dht.h>
#include <LiquidCrystal.h>
#include <SdFat.h>

SdFat sd;
SdFile myFile;
const int chipSelect = 8;

LiquidCrystal lcd(7, 6, 5, 4, 3, 2);
dht DHT;

#define DHT22_PIN 9

void setup()
{
  lcd.begin(16, 2);

  if (!sd.init(SPI_HALF_SPEED, chipSelect)) sd.initErrorHalt();

  if (!myFile.open("TempLog.txt", O_RDWR | O_CREAT | O_AT_END)) {
    sd.errorHalt("opening test.txt for write failed");
  }
  myFile.println();
  myFile.println("Temperature (F), Humidity (%)");
  myFile.close();
}

Done uploading.
Binary sketch size: 15,706 bytes (of a 32,256 byte maximum)

27 Arduino Uno on COM4
```

LOGGING TO μ SD CARD (CONTINUED)

- Complete the void loop portion of the program.
- We will be taking the same readings from the sensor and printing to the LCD in the same exact way.
- We are adding an Open to a text file on the SD card and then we are printing a comma separated value for the temperature and humidity on a new line every 1 second.



```
lcdandtemplogging | Arduino 1.0.3
File Edit Sketch Tools Help

lcdandtemplogging $

void loop()
{
    // READ DATA
    int chk = DHT.read22(DHT22_PIN);
    float celsiustemp=DHT.temperature;
    float farentemp = (celsiustemp * 9.0)/ 5.0 + 32.0;

    lcd.clear();
    lcd.setCursor(0,0);

    // DISPLAY DATA
    lcd.print("Temp: ");
    lcd.print(farentemp, 1);
    lcd.print(" F");
    lcd.setCursor(0, 1);
    lcd.print("Humidity: ");
    lcd.print(DHT.humidity, 1);
    lcd.print("%");

    // if the file opened okay, write to it:
    if (!sd.init(SPI_HALF_SPEED, chipSelect)) sd.initErrorHalt();

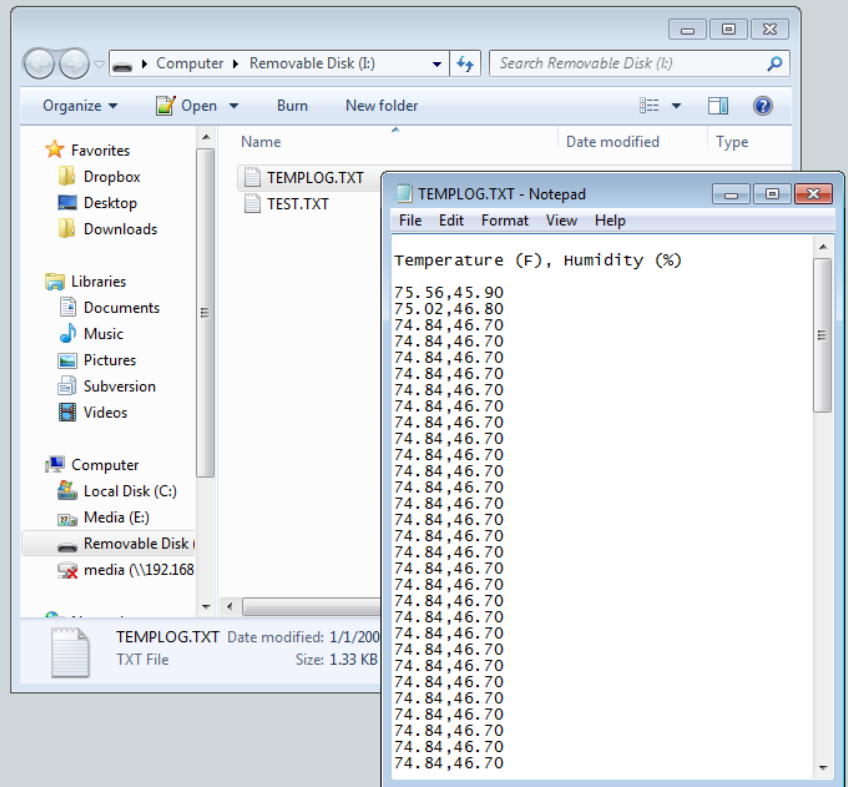
    if (!myFile.open("Templog.txt", O_RDWR | O_CREAT | O_AT_END)) {
        sd.errorHalt("opening test.txt for write failed");
    }

    myFile.println();
    myFile.print(farentemp);
    myFile.print(",");
    myFile.print(DHT.humidity);
    myFile.close();

    delay(1000);
}
```

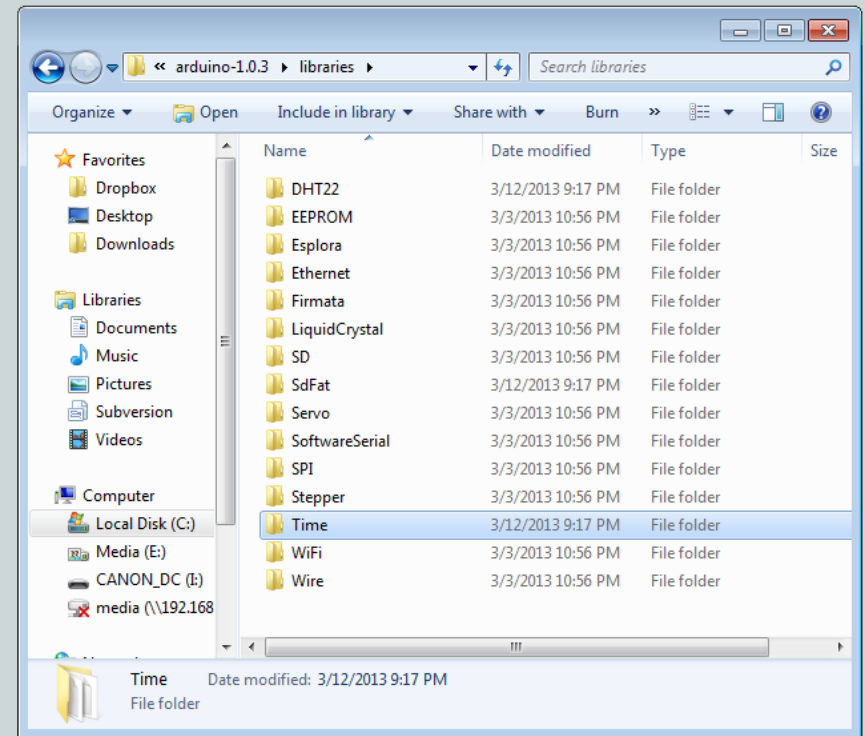
LOGGING TO μ SD CARD (CONTINUED)

- Upload to the Arduino with the SD card in the shield.
- After a few seconds (enough for a few readings) unplug the Arduino USB (so that it isn't writing to the card anymore) and remove the SD card.
- Open it on your computer and if it worked you should see something like this!



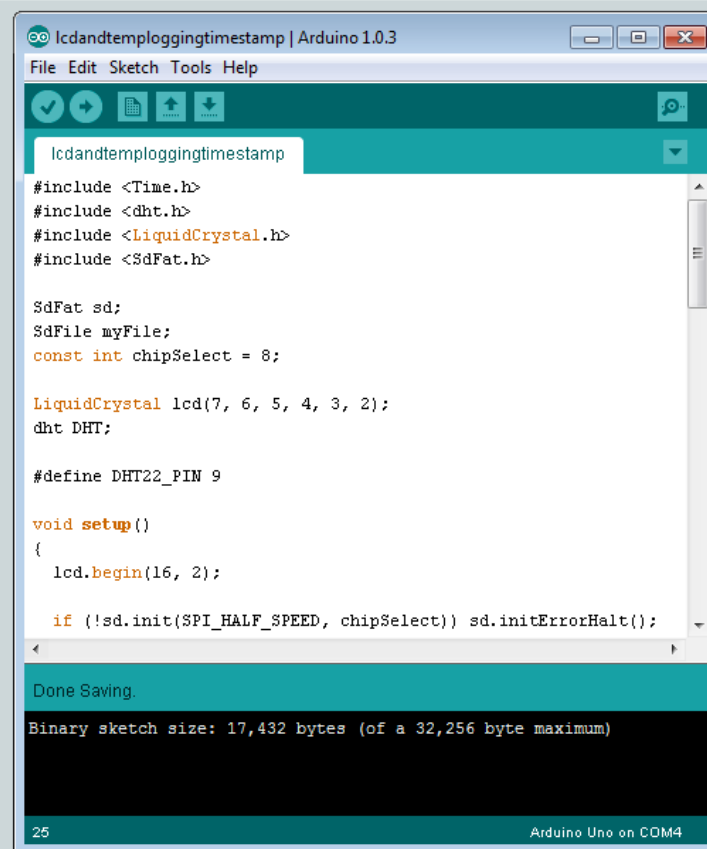
LOGGING TO μ SD CARD (CONTINUED)

- The problem is, there is no way to keep track of time easily on the Arduino since it doesn't have a battery to remember the time.
- With the last few steps, we will add a timestamp to this data so that it is a bit more useful when plotting or analyzing
- Copy the “Time” folder into the Arduino libraries folder



LOGGING TO μ SD CARD (CONTINUED)

To add the timestamp, first add the `#include<Time.h>` to the top of the program.



```
lcdandtemploggingtimestamp
#include <Time.h>
#include <dht.h>
#include <LiquidCrystal.h>
#include <SdFat.h>

SdFat sd;
SdFile myFile;
const int chipSelect = 8;

LiquidCrystal lcd(7, 6, 5, 4, 3, 2);
dht DHT;

#define DHT22_PIN 9

void setup()
{
  lcd.begin(16, 2);

  if (!sd.init(SPI_HALF_SPEED, chipSelect)) sd.initErrorHalt();
}
```

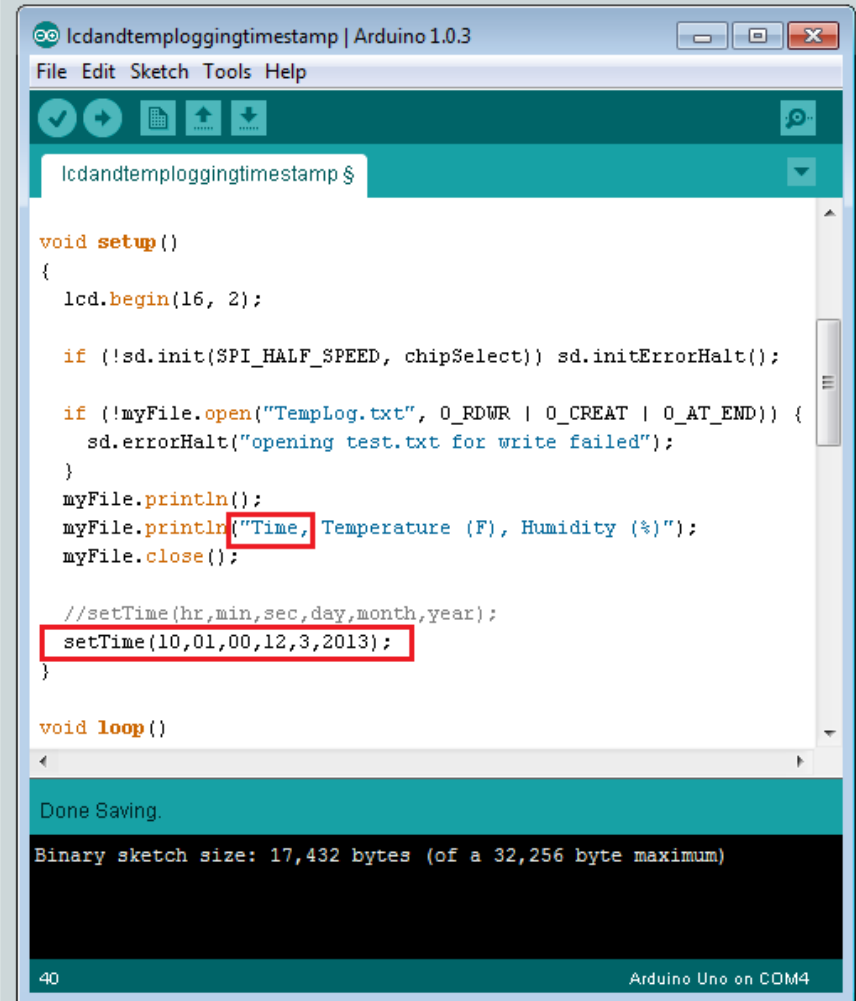
Done Saving.

Binary sketch size: 17,432 bytes (of a 32,256 byte maximum)

25 Arduino Uno on COM4

LOGGING TO μ SD CARD (CONTINUED)

- In void setup make 2 changes
- First add a “Time,” to the header of the text file
- Second we need to initialize the time for the Arduino. As soon as you click Upload, it will set this time to the Arduino. This means if you unplug and plug the Arduino back in, it will reinitialize this as the time even if it is not right.
- You will need to modify this and reupload later



```
lcdandtemploggingtimestamp | Arduino 1.0.3
File Edit Sketch Tools Help

lcdandtemploggingtimestamp $

void setup()
{
  lcd.begin(16, 2);

  if (!sd.init(SPI_HALF_SPEED, chipSelect)) sd.initErrorHalt();

  if (!myFile.open("TempLog.txt", O_RDWR | O_CREAT | O_AT_END)) {
    sd.errorHalt("opening test.txt for write failed");
  }
  myFile.println();
  myFile.println("Time, Temperature (F), Humidity (%)");
  myFile.close();

  //setTime(hr,min,sec,day,month,year);
  setTime(10,01,00,12,3,2013);
}

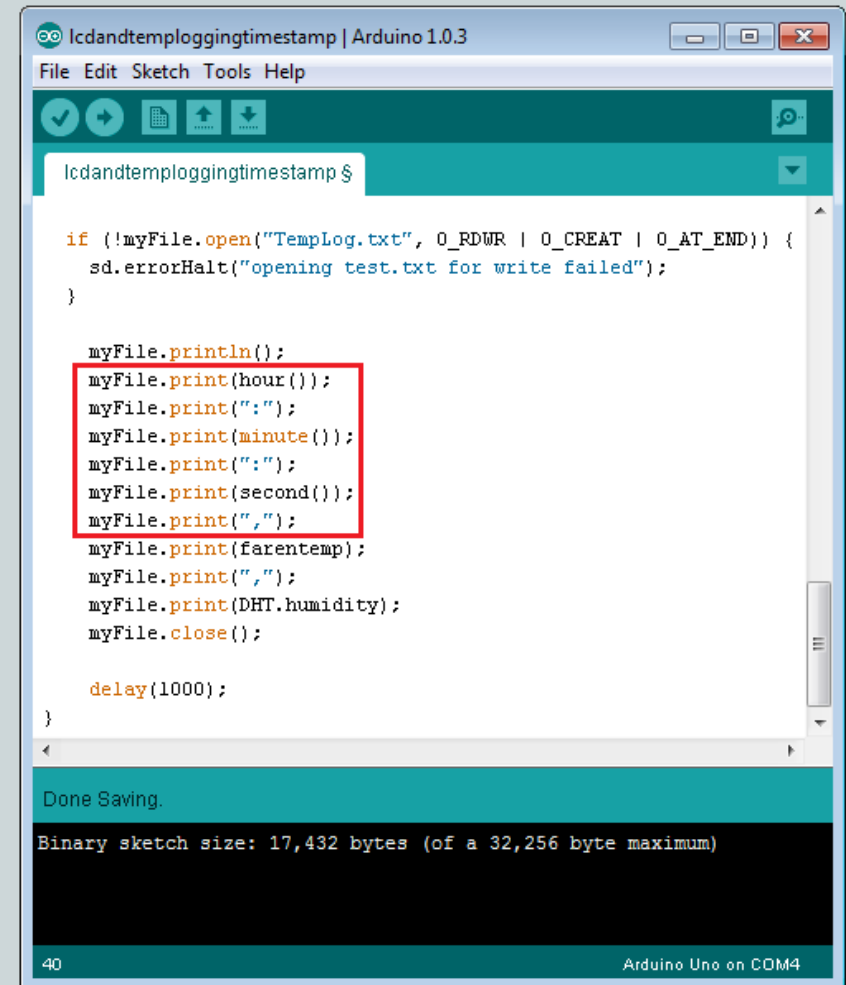
void loop()

Done Saving.
Binary sketch size: 17,432 bytes (of a 32,256 byte maximum)

40 Arduino Uno on COM4
```

LOGGING TO μ SD CARD (CONTINUED)

- In void loop we need to add the time to the data being logged.
- After the newline we want to print the hour minute and second value followed by a comma.
- This will come before the temp and the humidity to match the header.
- These will update when the Arduino is on based on the initialized time performed by setTime in void setup



```
lcdandtemploggingtimestamp | Arduino 1.0.3
File Edit Sketch Tools Help

lcdandtemploggingtimestamp $

if (!myFile.open("TempLog.txt", O_RDWR | O_CREAT | O_AT_END)) {
  sd.errorHalt("opening test.txt for write failed");
}

myFile.println();
myFile.print(hour());
myFile.print(":");
myFile.print(minute());
myFile.print(":");
myFile.print(second());
myFile.print(",");
myFile.print(farentemp);
myFile.print(",");
myFile.print(DHT.humidity);
myFile.close();

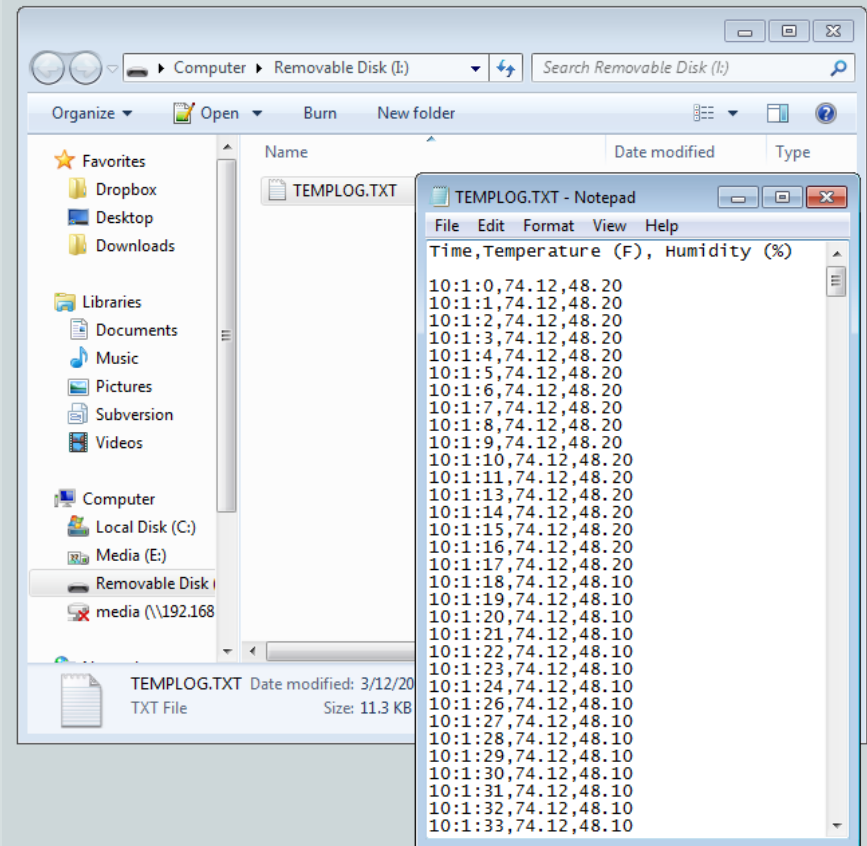
delay(1000);
}

Done Saving.
Binary sketch size: 17,432 bytes (of a 32,256 byte maximum)

40 Arduino Uno on COM4
```


LOGGING TO μ SD CARD (CONTINUED)

- Now adjust the setTime in void setup to be a minute from now
- Upload the program and wait for it to start running.
- After a bit unplug the USB and plug the SD card into your computer
- You should get something like this!
- If you plug the USB back in with the card in, you will want to set the time again and upload again



DONE!

You did it! Now you have a thing that does a thing!