

HiBall Balloon Payload Workshop

Arduino Introduction





Arduino Driving Lessons

- A. Arduino IDE**
- B. Arduino Overview**
- C. Arduino Communication**
- D. Blink an Led, Change the World**



Download Arduino IDE:

• What's an IDE?

- Integrated Development Environment

- Word processor for source code
- All development tools in one place
 - Syntax Highlighting
 - Autocomplete
 - Analyzers
 - Debugging
- Reconfigurable to each developers style

```
void setup() {  
  // put your setup code here, to  
  
  Serial.begin(9600);  
  
  // setup the LED Visual Display  
  pinMode(3, OUTPUT); //Ar  
  pinMode(4, OUTPUT); //In  
  pinMode(5, OUTPUT); //Ex  
  pinMode(6, OUTPUT); //Hu  
  pinMode(7, OUTPUT); //Pr  
  pinMode(9, OUTPUT); //Ac  
  
  // turn on Arduino LED  
  digitalWrite(3, HIGH); //  
  
  // Print Column Headers  
  Serial.println("Time,Temp  
}
```



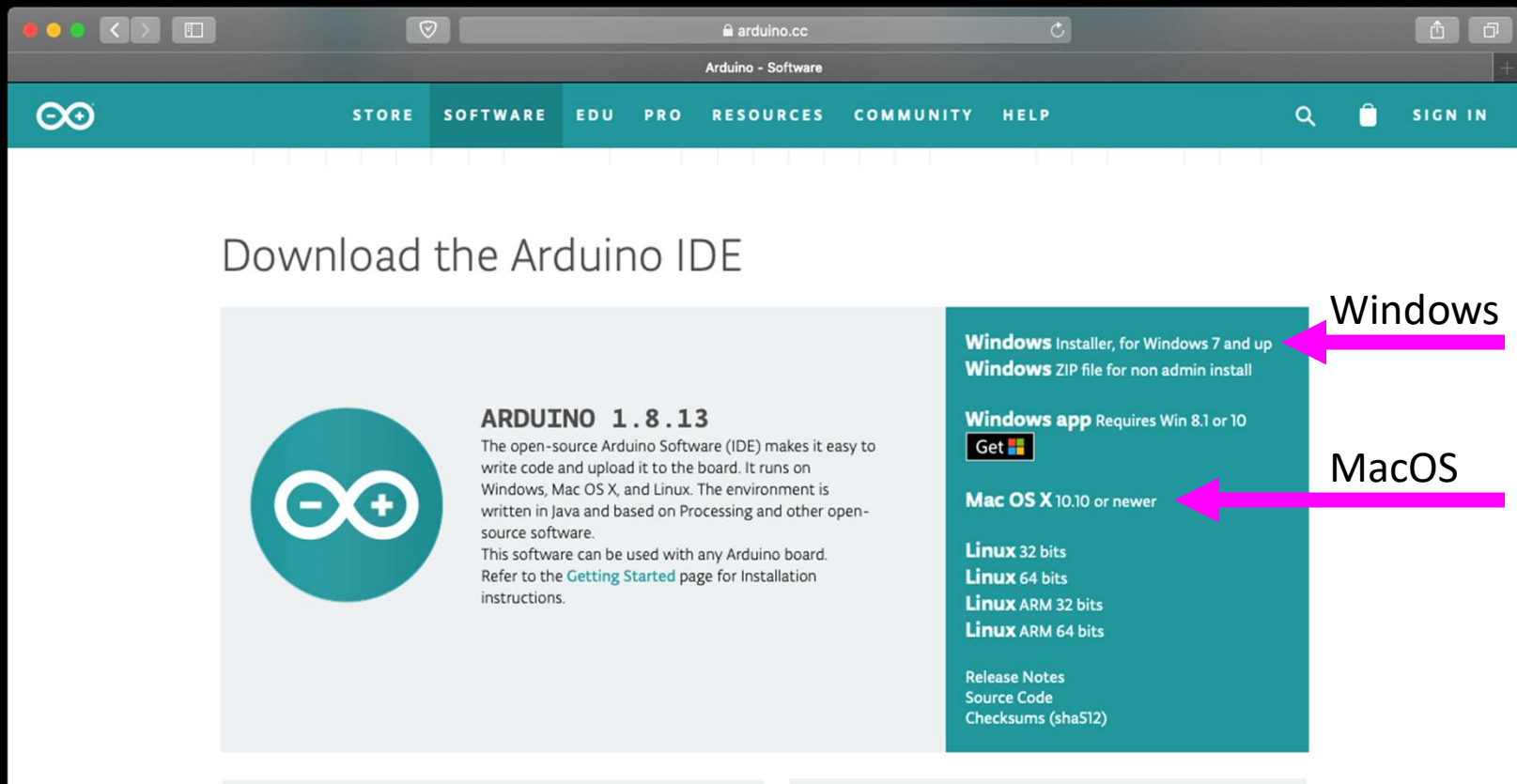
Download Arduino IDE:

```
void setup() {  
  // put your setup code here, to run once:  
  
  Serial.begin(9600);  
  
  // setup the LED Visual Display  
  pinMode(3, OUTPUT); //Arduino on  
  pinMode(4, OUTPUT); //Internal Temp  
  pinMode(5, OUTPUT); //External Temp  
  pinMode(6, OUTPUT); //Humidity  
  pinMode(7, OUTPUT); //Pressure  
  pinMode(9, OUTPUT); //Accels  
  
  // turn on Arduino LED  
  digitalWrite(3, HIGH); // Leave on while  
  
  // Print Column Headers  
  Serial.println("Time,Temp1F,Temp2F,RH,Pres  
}
```

```
void setup() {  
  // put your setup code here, to run once:  
  
  Serial.begin(9600);  
  
  // setup the LED Visual Display  
  pinMode(3, OUTPUT); //Arduino on  
  pinMode(4, OUTPUT); //Internal Temp  
  pinMode(5, OUTPUT); //External Temp  
  pinMode(6, OUTPUT); //Humidity  
  pinMode(7, OUTPUT); //Pressure  
  pinMode(9, OUTPUT); //Accels  
  
  // turn on Arduino LED  
  digitalWrite(3, HIGH); // Leave on whi  
  
  // Print Column Headers  
  Serial.println("Time,Temp1F,Temp2F,RH,P  
}
```

Download Arduino IDE:

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

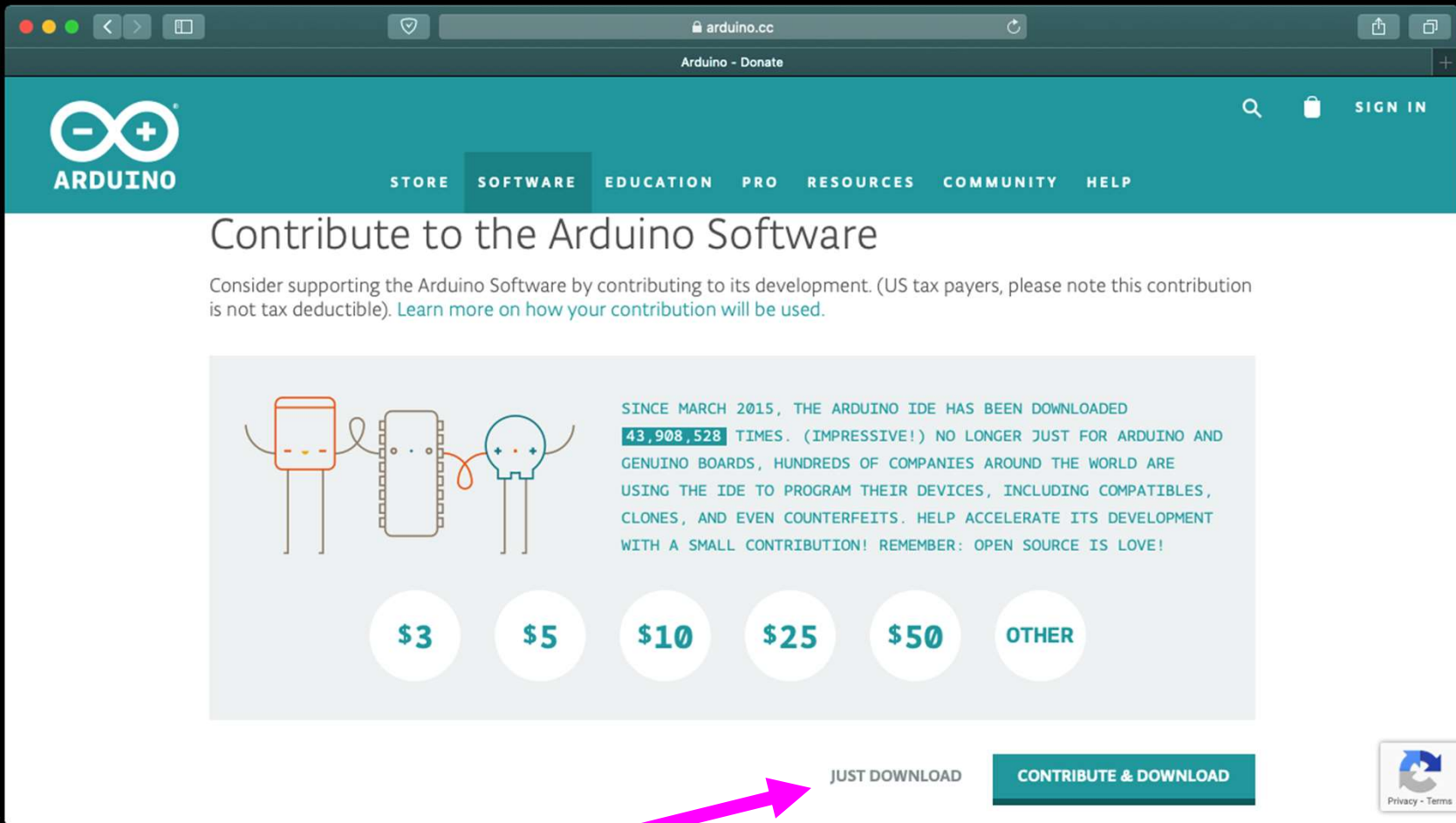


The screenshot shows the Arduino Software download page. The page has a teal header with the Arduino logo and navigation links: STORE, SOFTWARE, EDU, PRO, RESOURCES, COMMUNITY, and HELP. A search icon, a shopping cart icon, and a SIGN IN link are on the right. The main content area is white and features the heading "Download the Arduino IDE". Below this heading is a large teal circle with the Arduino logo. To the right of the logo, the text reads: **ARDUINO 1.8.13**
The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.
This software can be used with any Arduino board. Refer to the [Getting Started](#) page for Installation instructions.

On the right side of the page, there is a teal box containing the following links: **Windows** Installer, for Windows 7 and up
Windows ZIP file for non admin install
Windows app Requires Win 8.1 or 10
Get [Windows logo]
Mac OS X 10.10 or newer
Linux 32 bits
Linux 64 bits
Linux ARM 32 bits
Linux ARM 64 bits
Release Notes
Source Code
Checksums (sha512)

Two pink arrows point to the "Windows" and "Mac OS X" links. The word "Windows" is written in black text above the top arrow, and "MacOS" is written in black text above the bottom arrow.

Download Arduino IDE:



arduino.cc

Arduino - Donate

ARDUINO

STORE SOFTWARE EDUCATION PRO RESOURCES COMMUNITY HELP

Contribute to the Arduino Software

Consider supporting the Arduino Software by contributing to its development. (US tax payers, please note this contribution is not tax deductible). [Learn more on how your contribution will be used.](#)

SINCE MARCH 2015, THE ARDUINO IDE HAS BEEN DOWNLOADED **43,908,528** TIMES. (IMPRESSIVE!) NO LONGER JUST FOR ARDUINO AND GENUINO BOARDS, HUNDREDS OF COMPANIES AROUND THE WORLD ARE USING THE IDE TO PROGRAM THEIR DEVICES, INCLUDING COMPATIBLES, CLONES, AND EVEN COUNTERFEITS. HELP ACCELERATE ITS DEVELOPMENT WITH A SMALL CONTRIBUTION! REMEMBER: OPEN SOURCE IS LOVE!

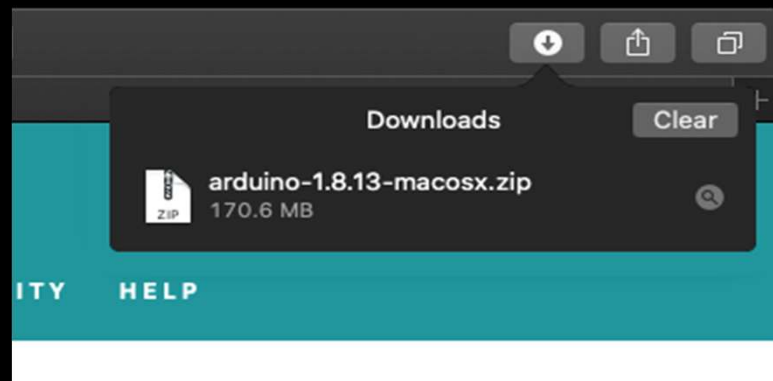
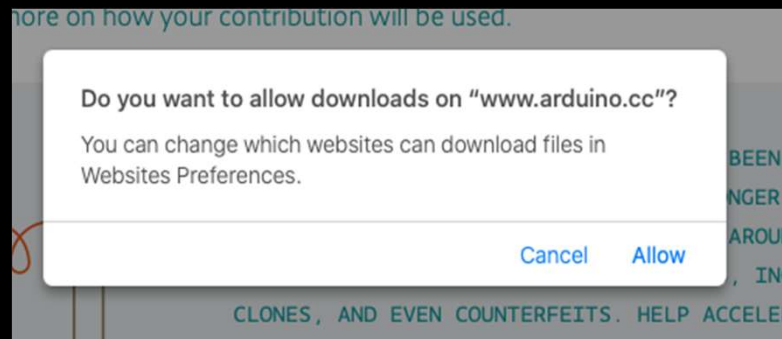
\$3 \$5 \$10 \$25 \$50 OTHER

JUST DOWNLOAD CONTRIBUTE & DOWNLOAD

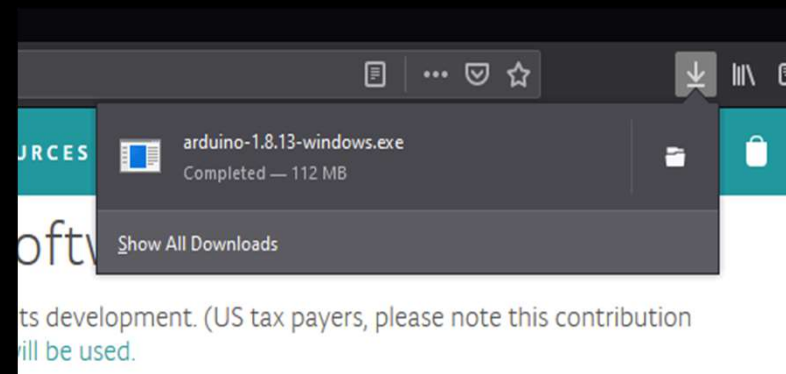
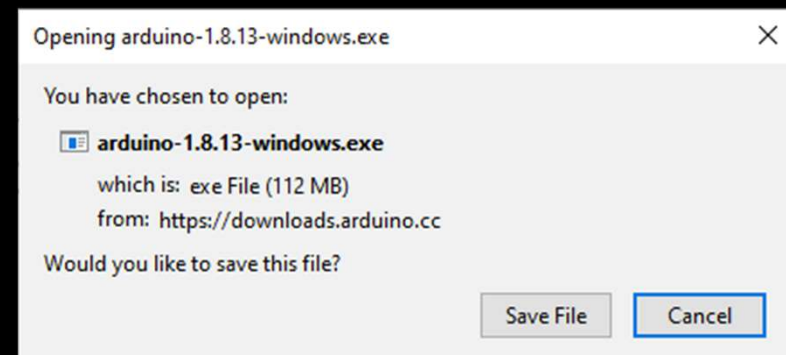
Privacy - Terms

Download Arduino IDE:

MacOS

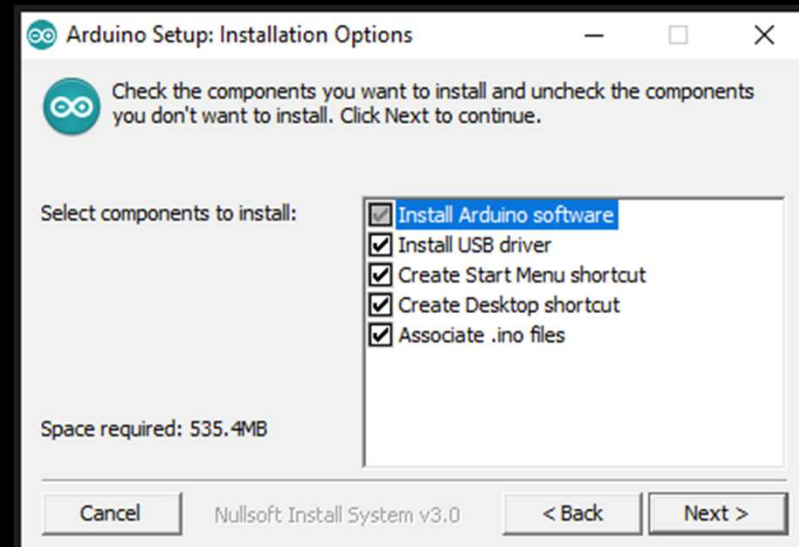
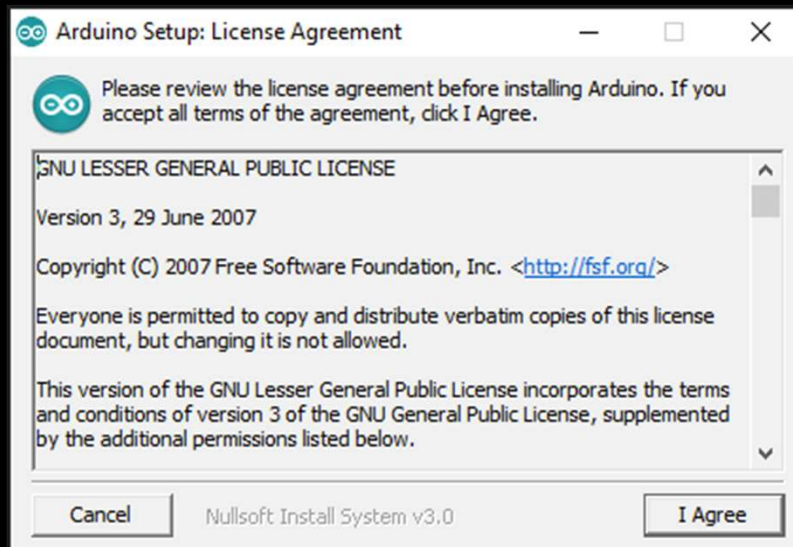


Windows



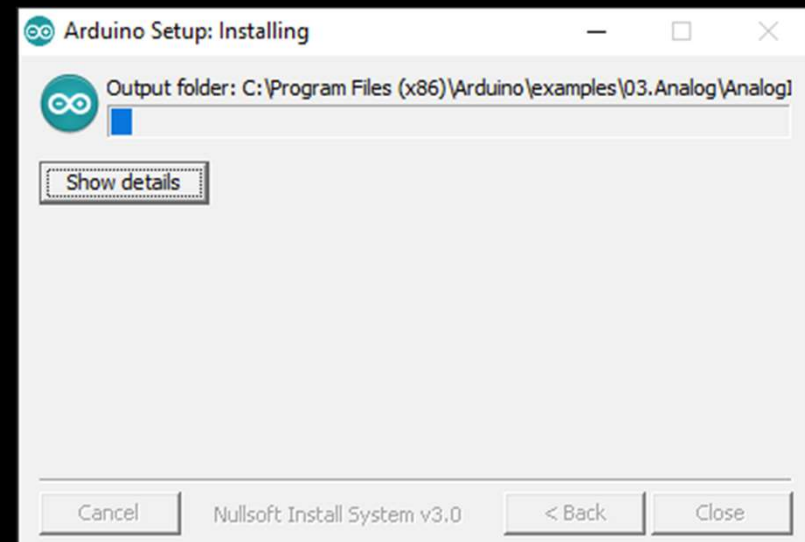
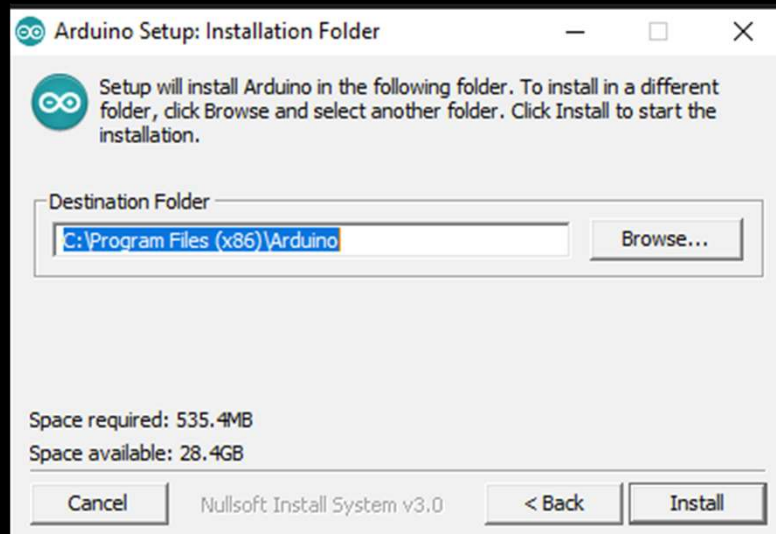
Download Arduino IDE:

Windows Installation



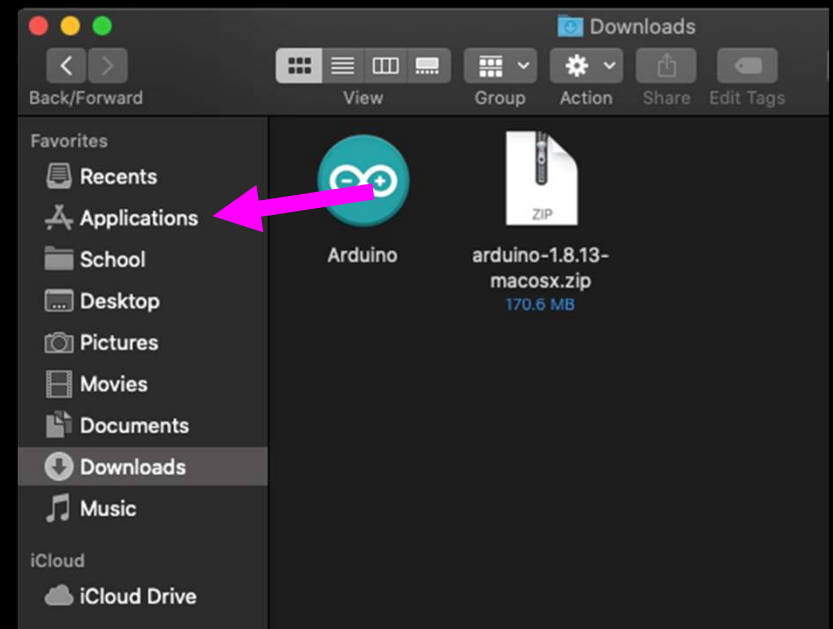
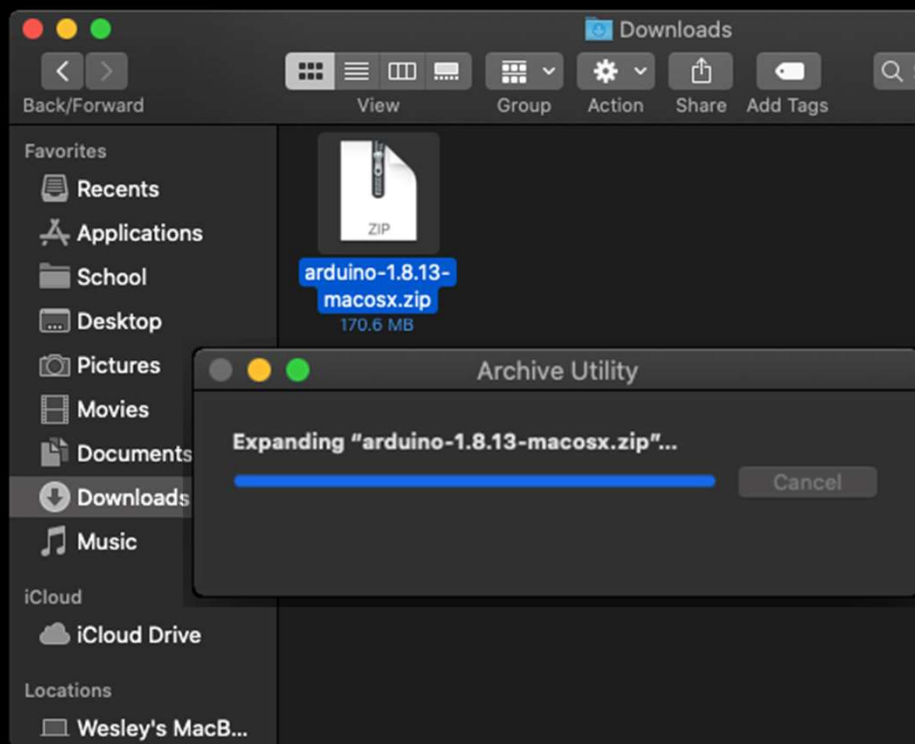
Download Arduino IDE:

Windows Installation (Cont.)



Download Arduino IDE:

Mac Installation



Download Arduino IDE:





Arduino Driving Lessons

- A. Download Arduino IDE**
- B. Arduino Overview**
- C. Arduino Communication**
- D. Blink an Led, Change the World**

Arduino Overview:

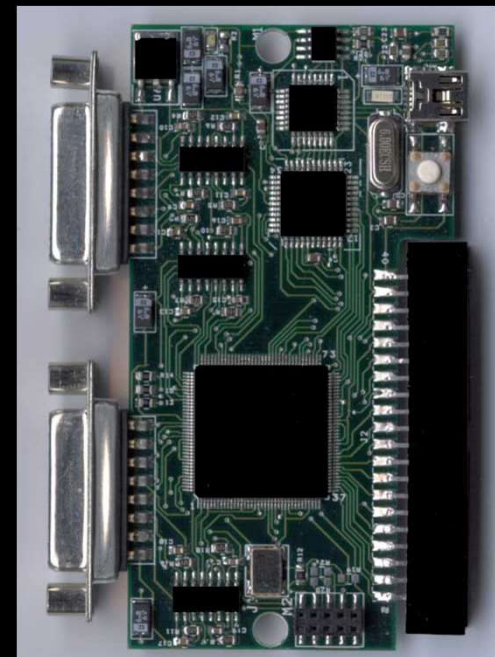
•General Purpose computer

- Usually has a human in the loop
- Can be reconfigured to do any number of tasks (excel, email, music)



•Embedded Systems

- Human input not required all the time
 - Takes specific inputs and computes outputs for a very specific application
 - Meets real-time goals
 - Heart monitor
 - Automatic braking systems (ABS)



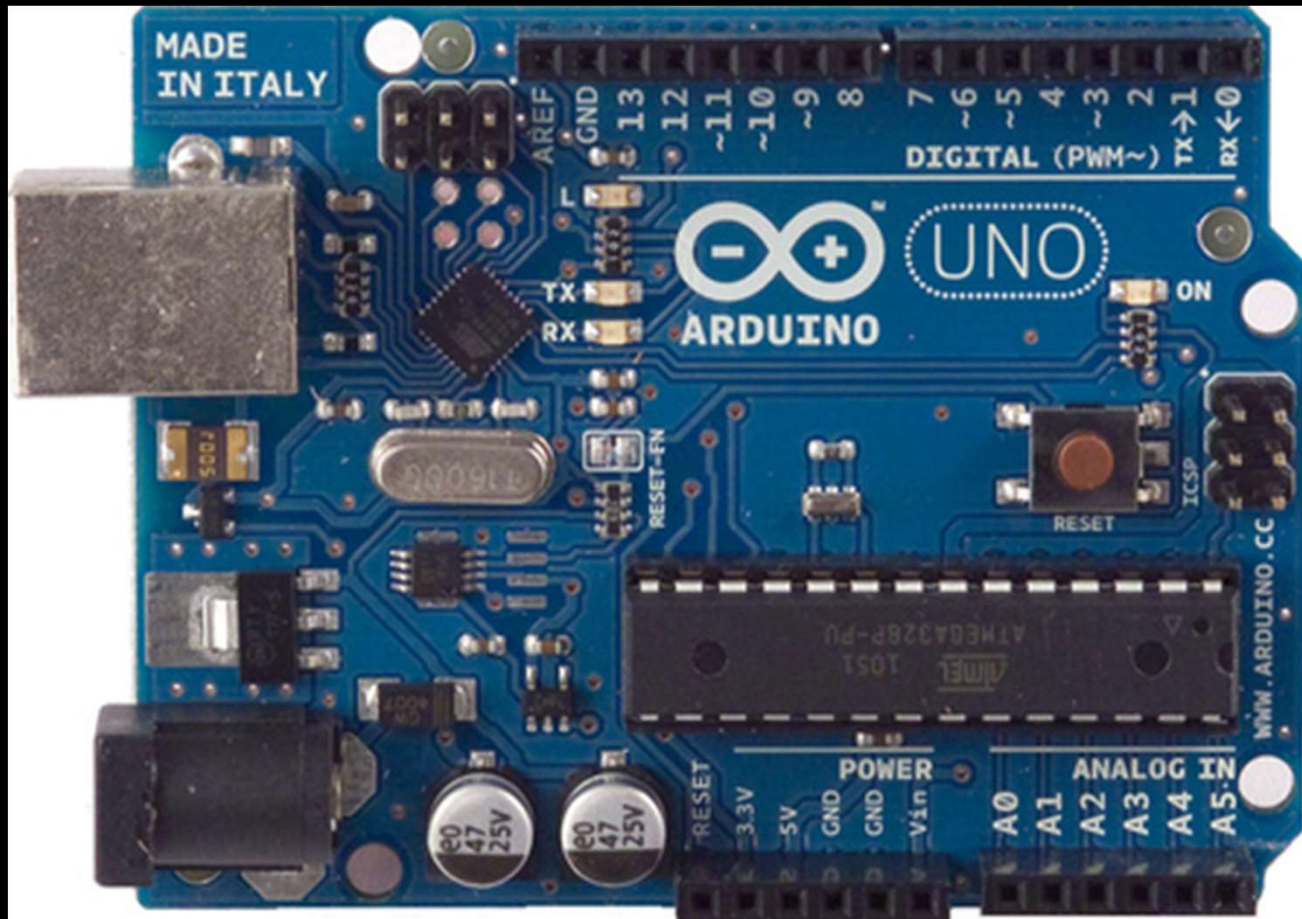
Arduino Overview:

- Arduino is an **embedded system**
- Board supports an **open source environment**, lots of assistance available online
- Extremely **modular**
- Types of Arduinos: **Uno**, Due, Mega...
- Each version has different capabilities
- Lots of **analog and digital I/O**

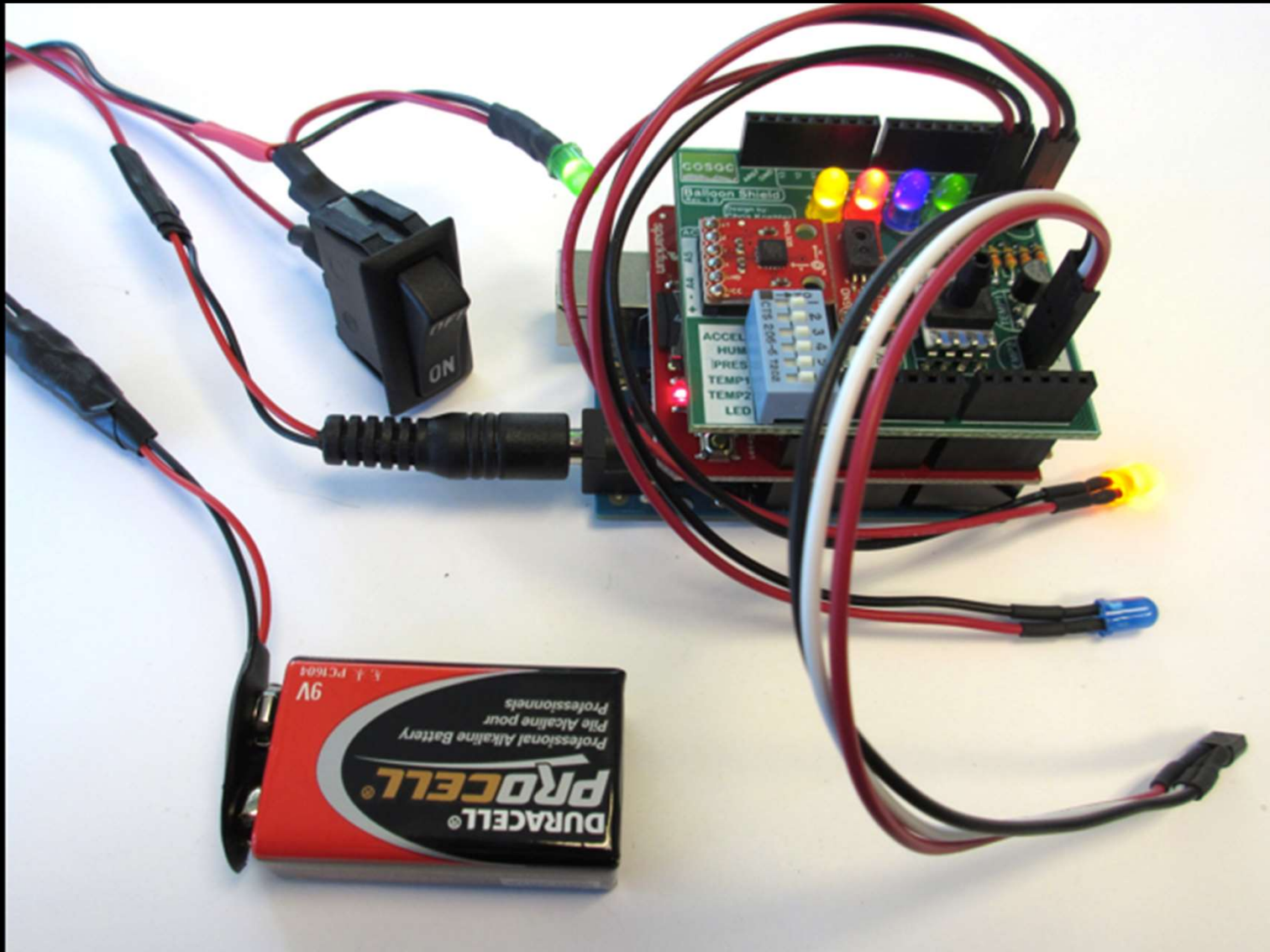


Arduino Overview:

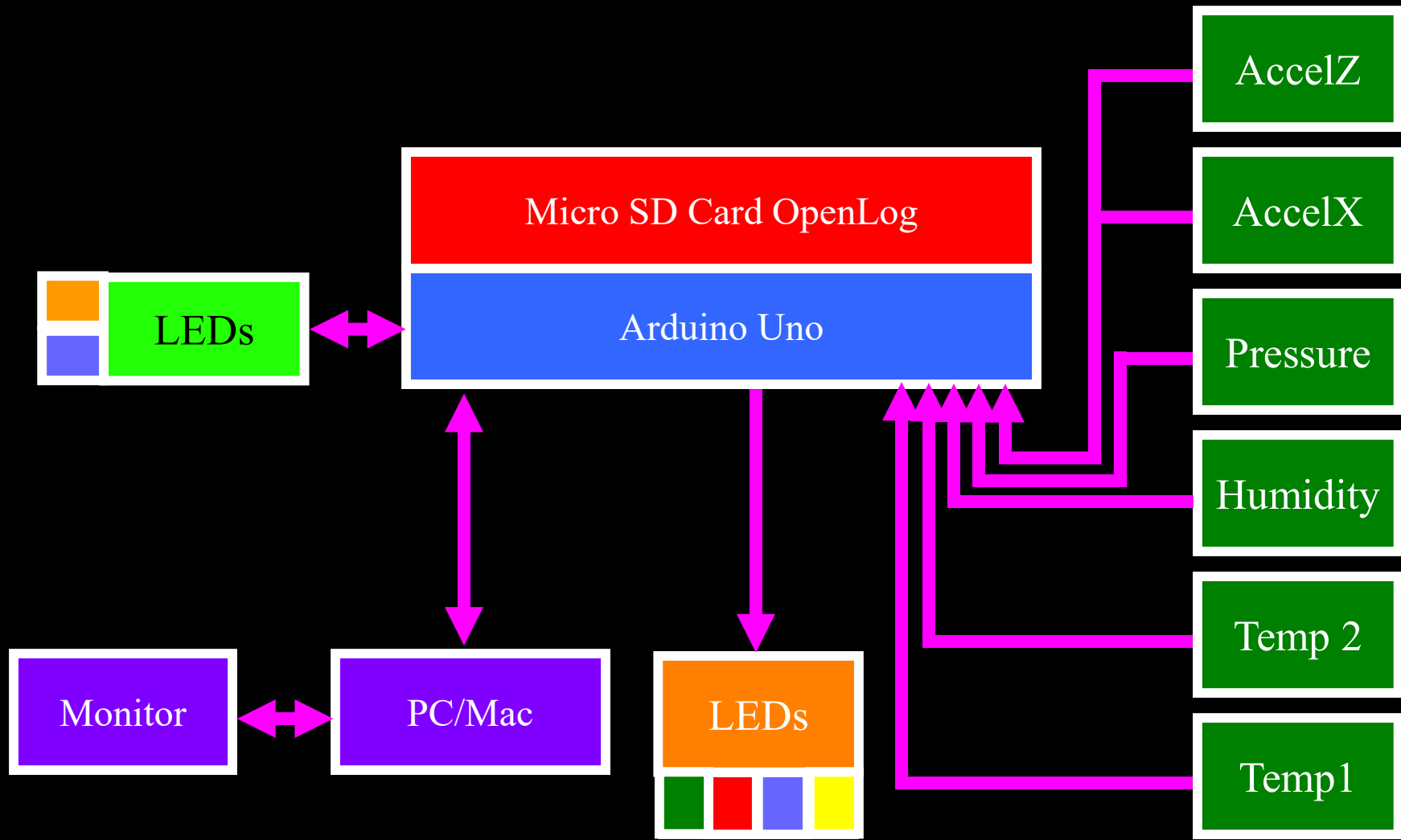
Arduino Uno Rev 3



Arduino Overview:



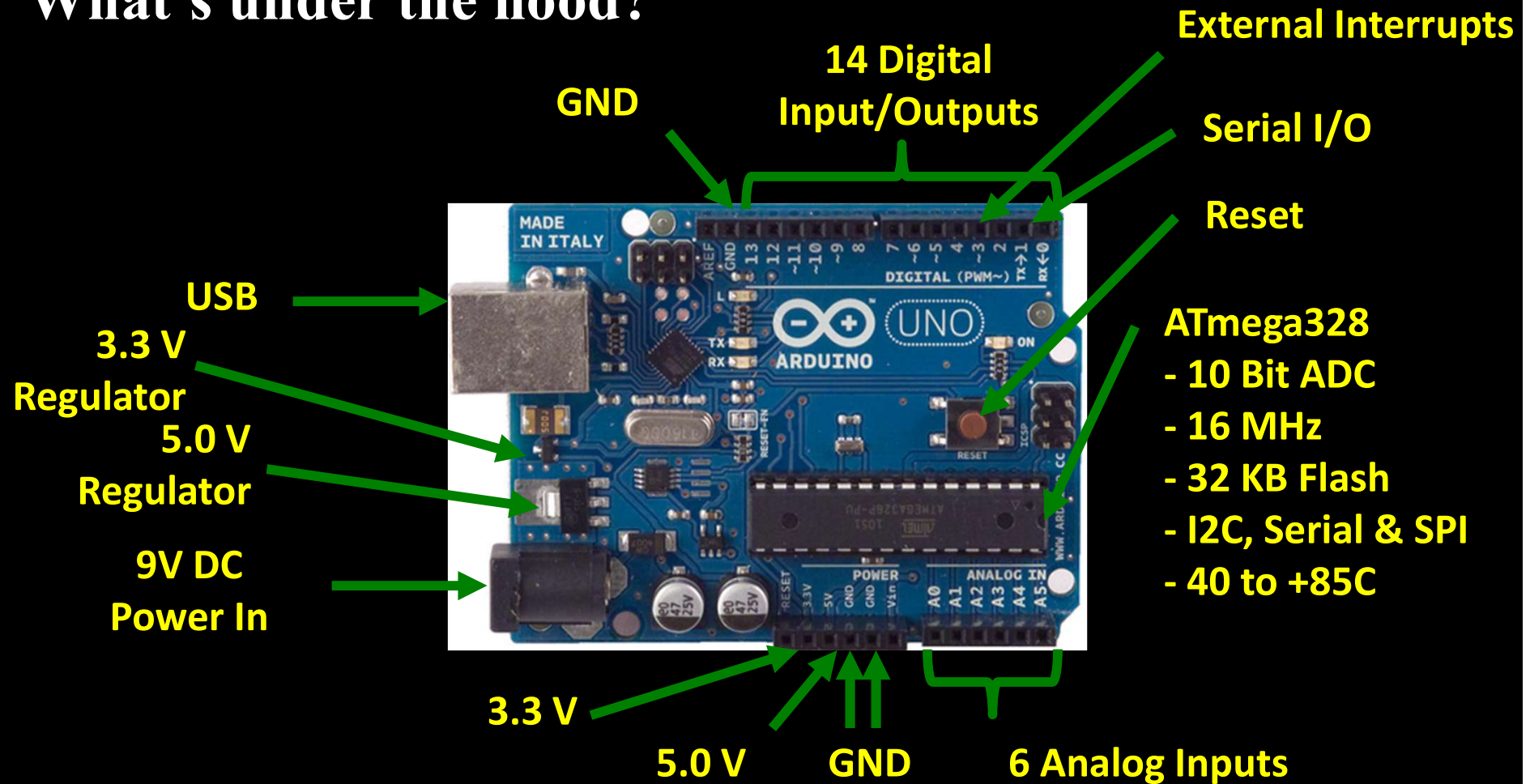
Arduino Overview:





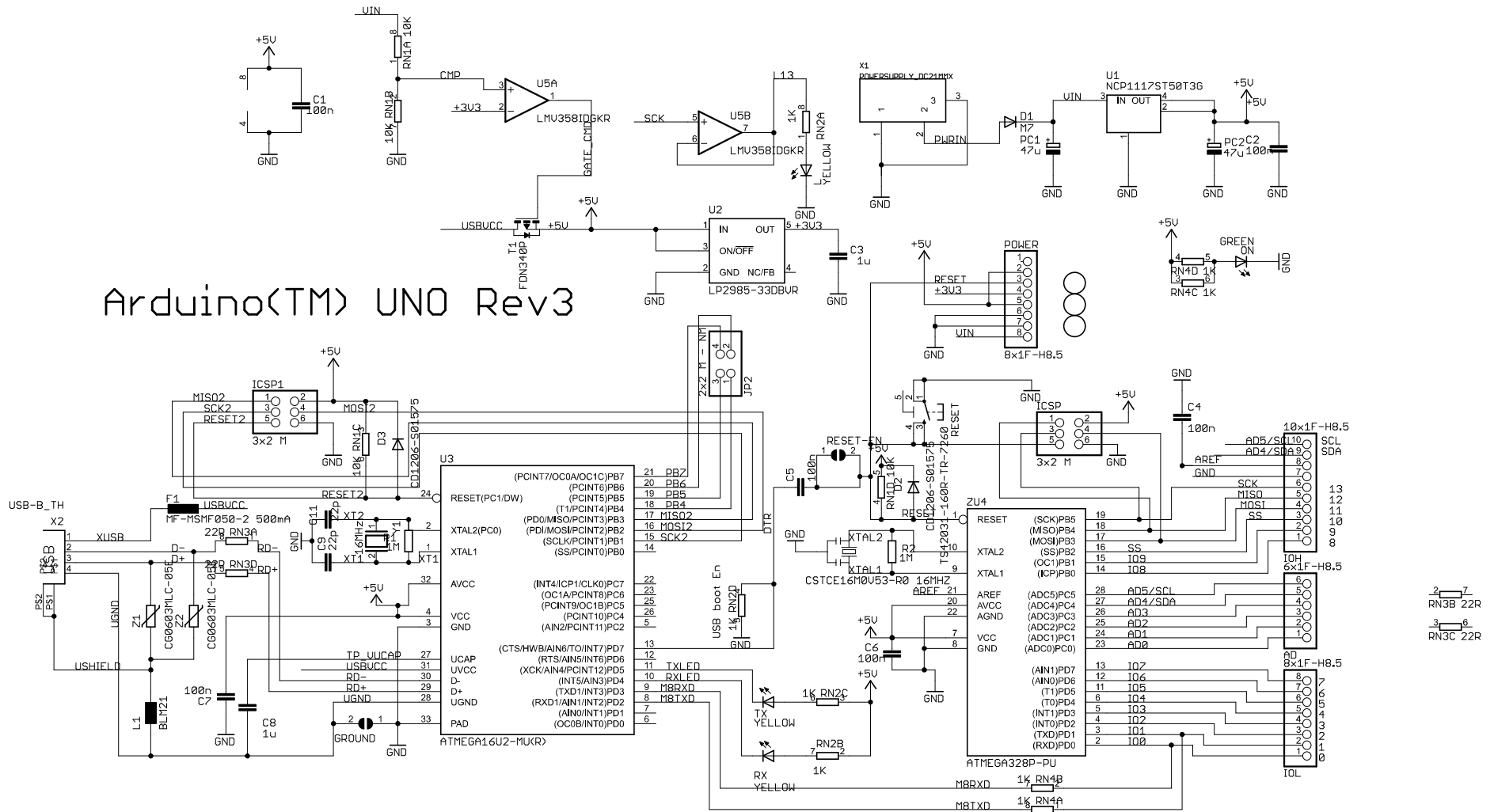
Arduino Overview:

What's under the hood?



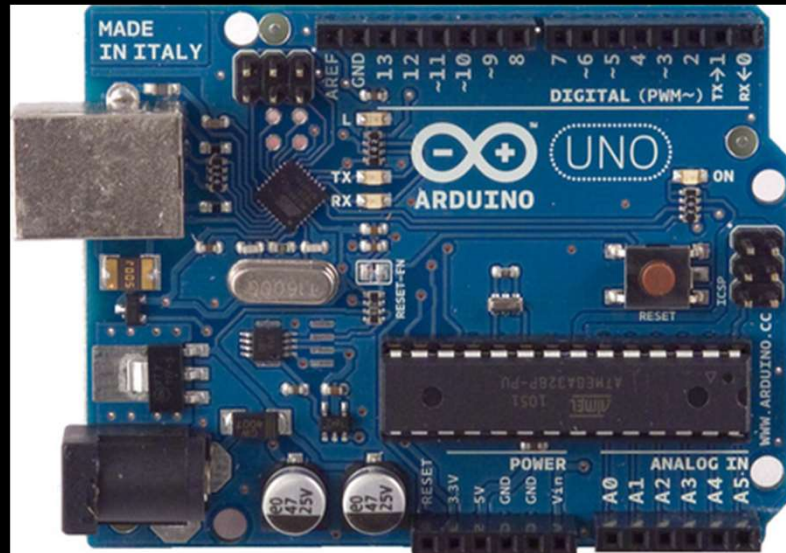
Arduino Overview:

Arduino(TM) UNO Rev3



Arduino Overview:

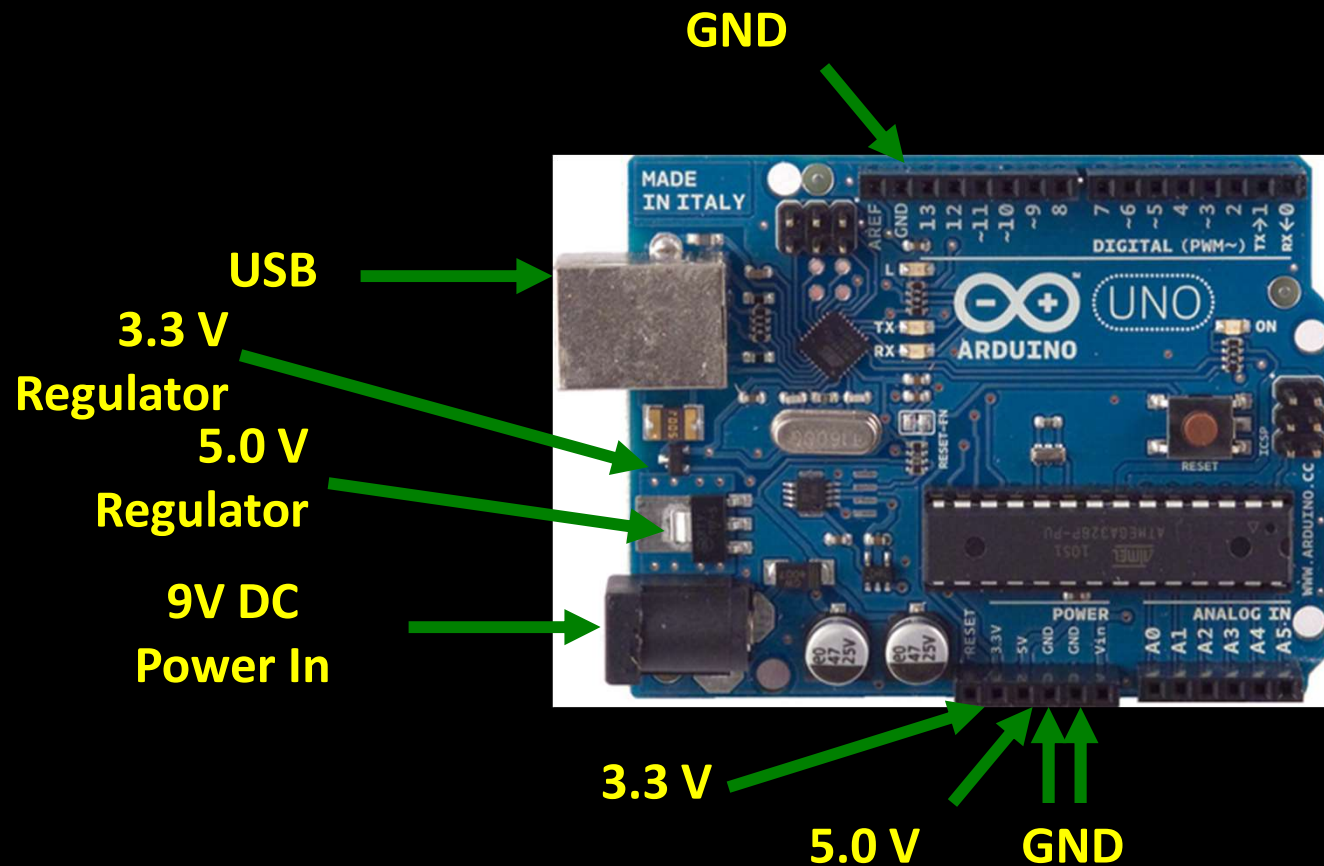
- So what does all that mean?



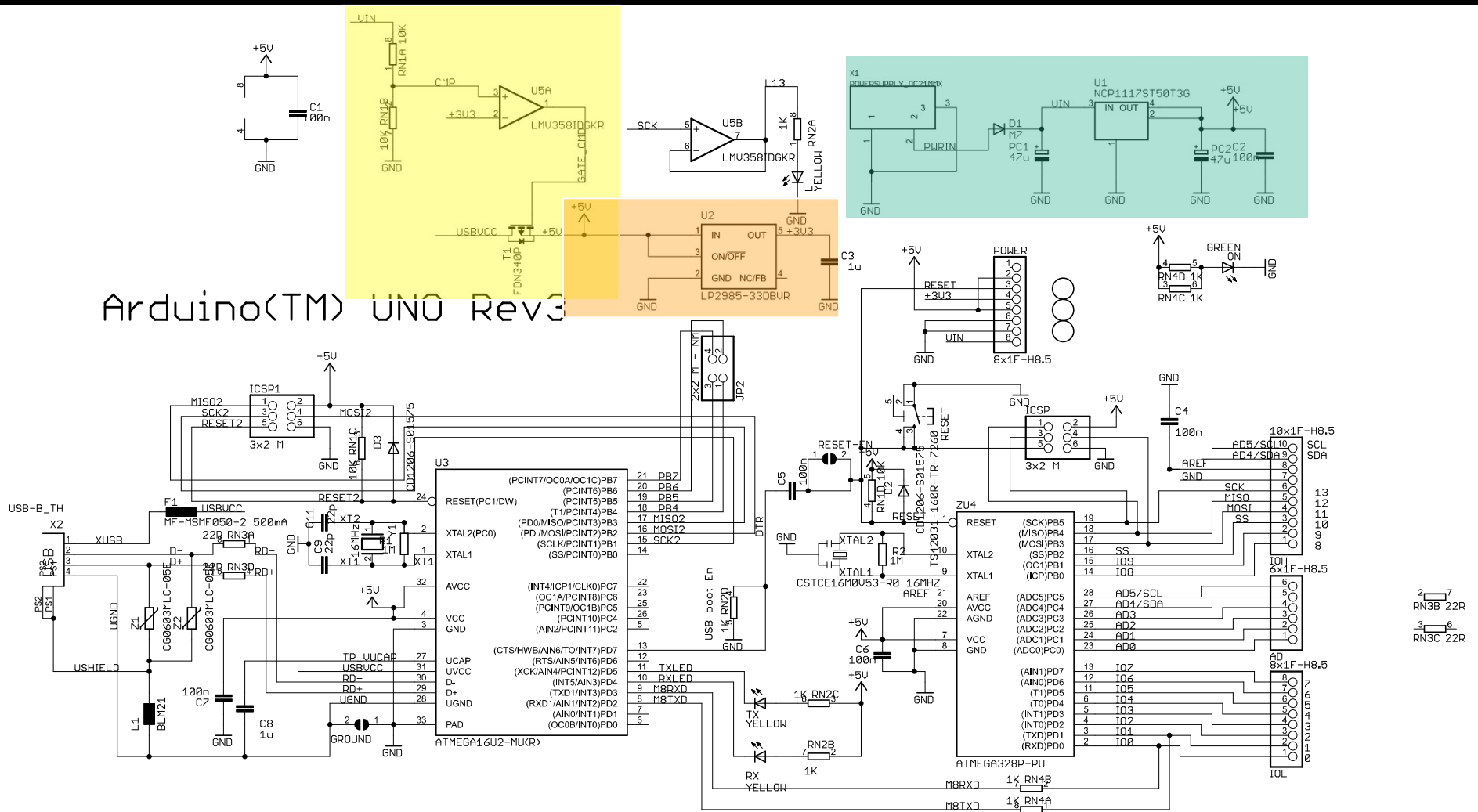


Arduino Overview:

The Easy Stuff...



Space Minor
UNIVERSITY OF COLORADO BOULDER

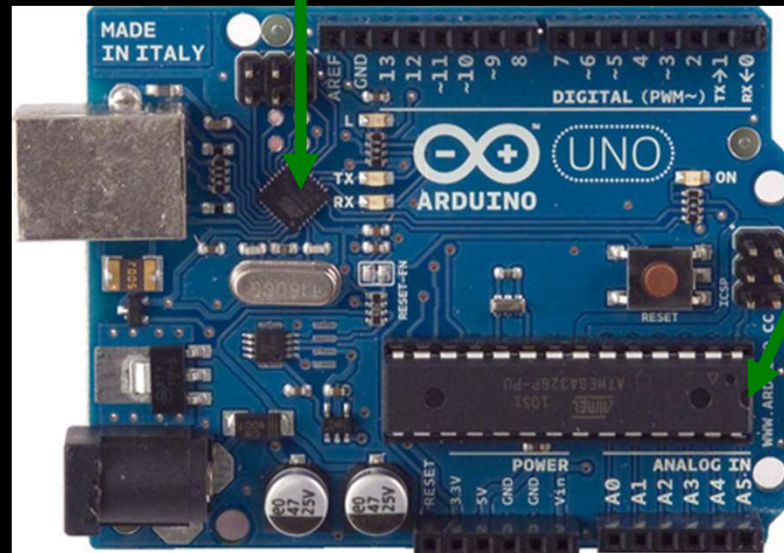


Arduino Overview:

The Chips...

ATmega16U

- Handles the USB interface to the computer
- We don't program this one

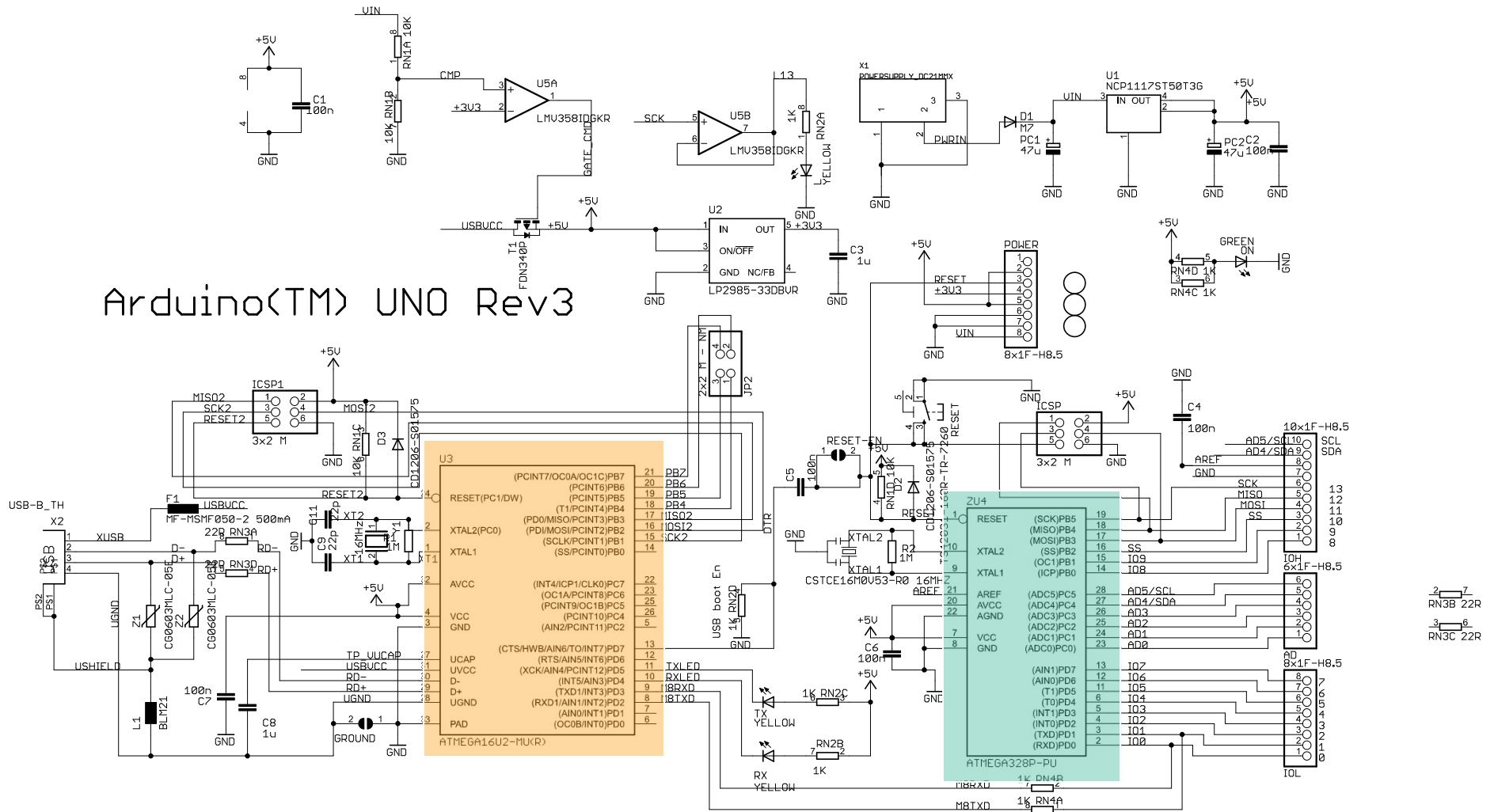


ATmega328

- 10 Bit ADC
- 16 MHz
- 32 KB Flash
- I2C & SPI
- 40 to +85C

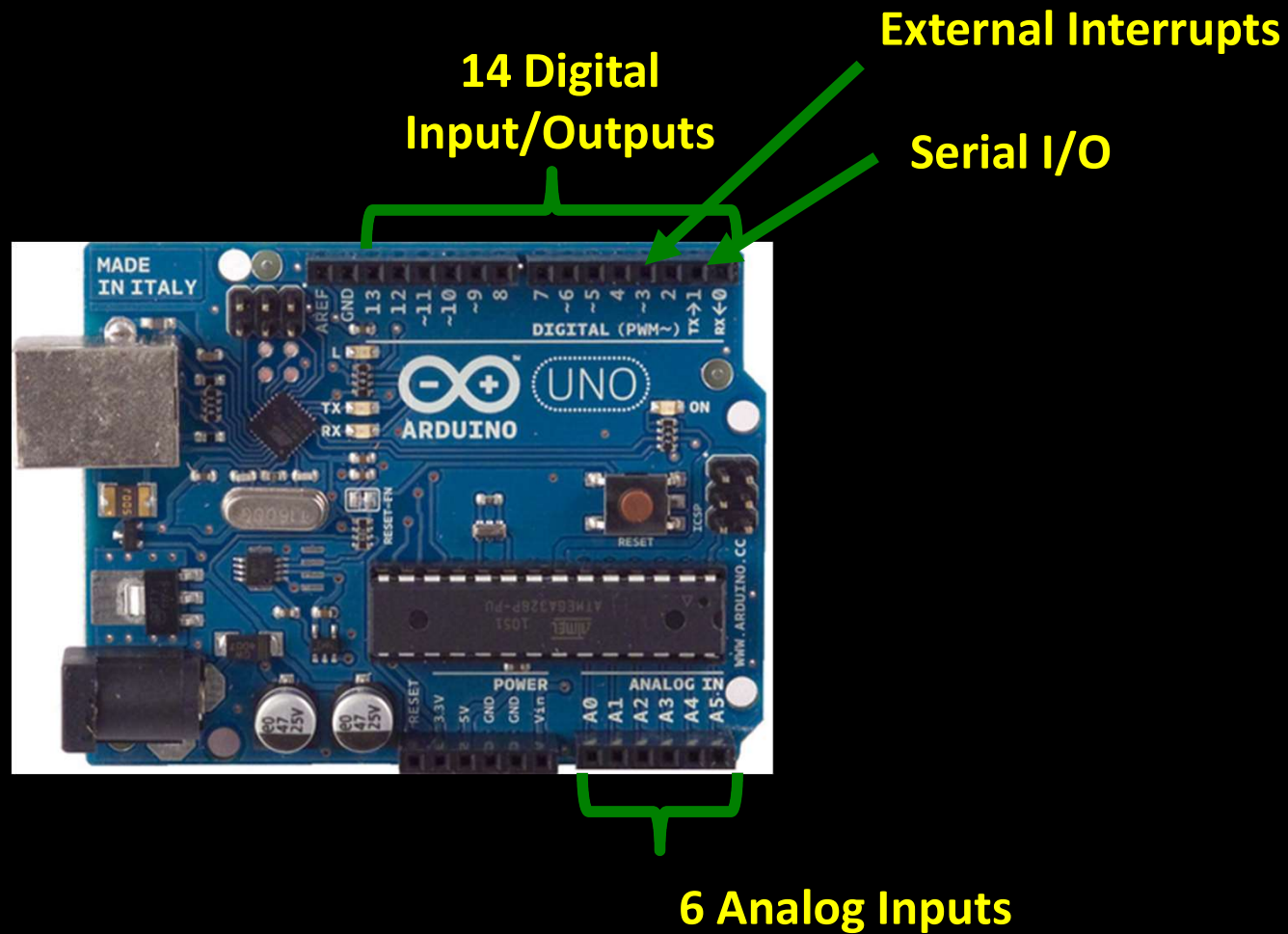
Arduino Overview:

Arduino(TM) UNO Rev3



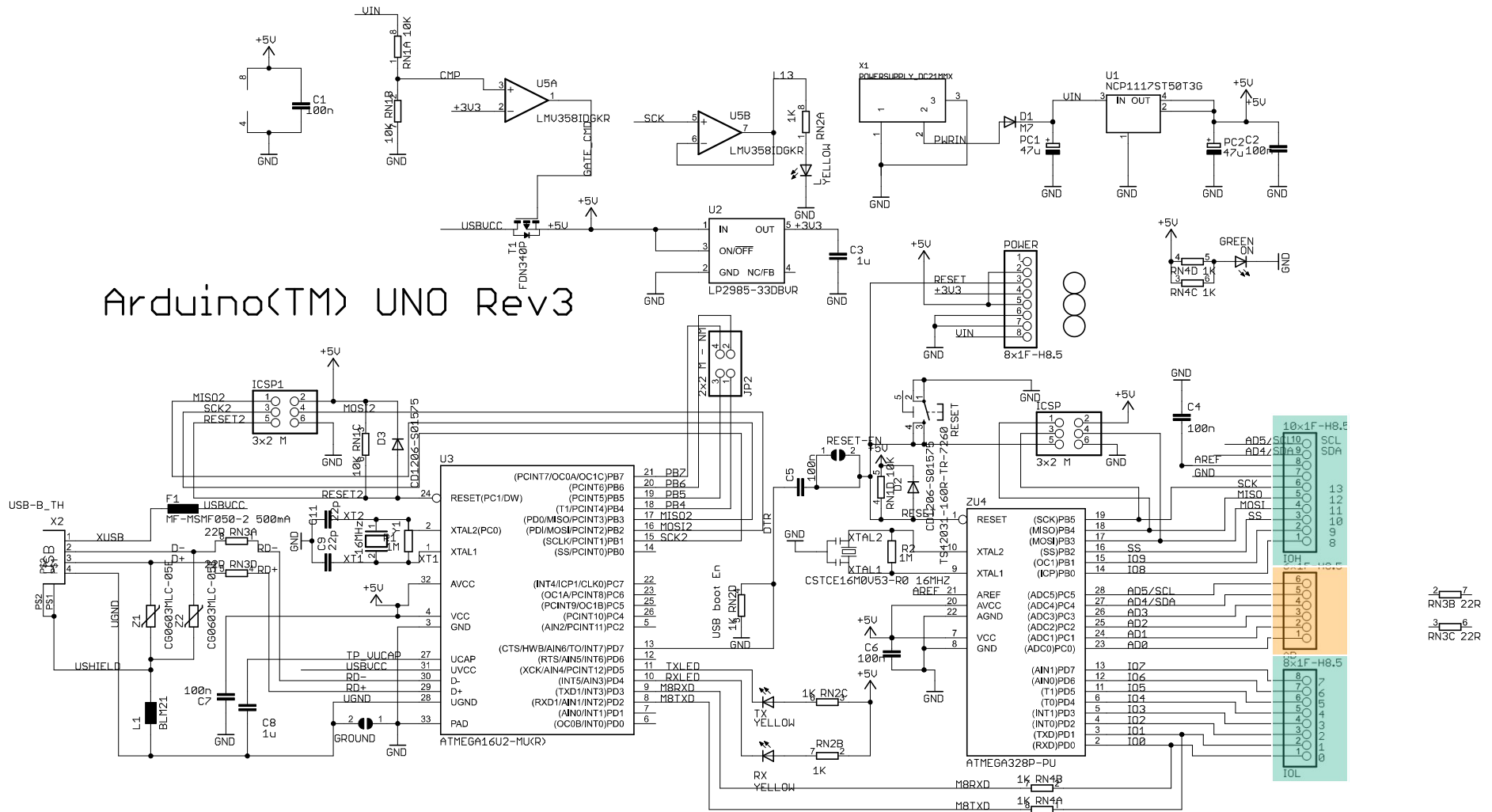
Arduino Overview:

Other...

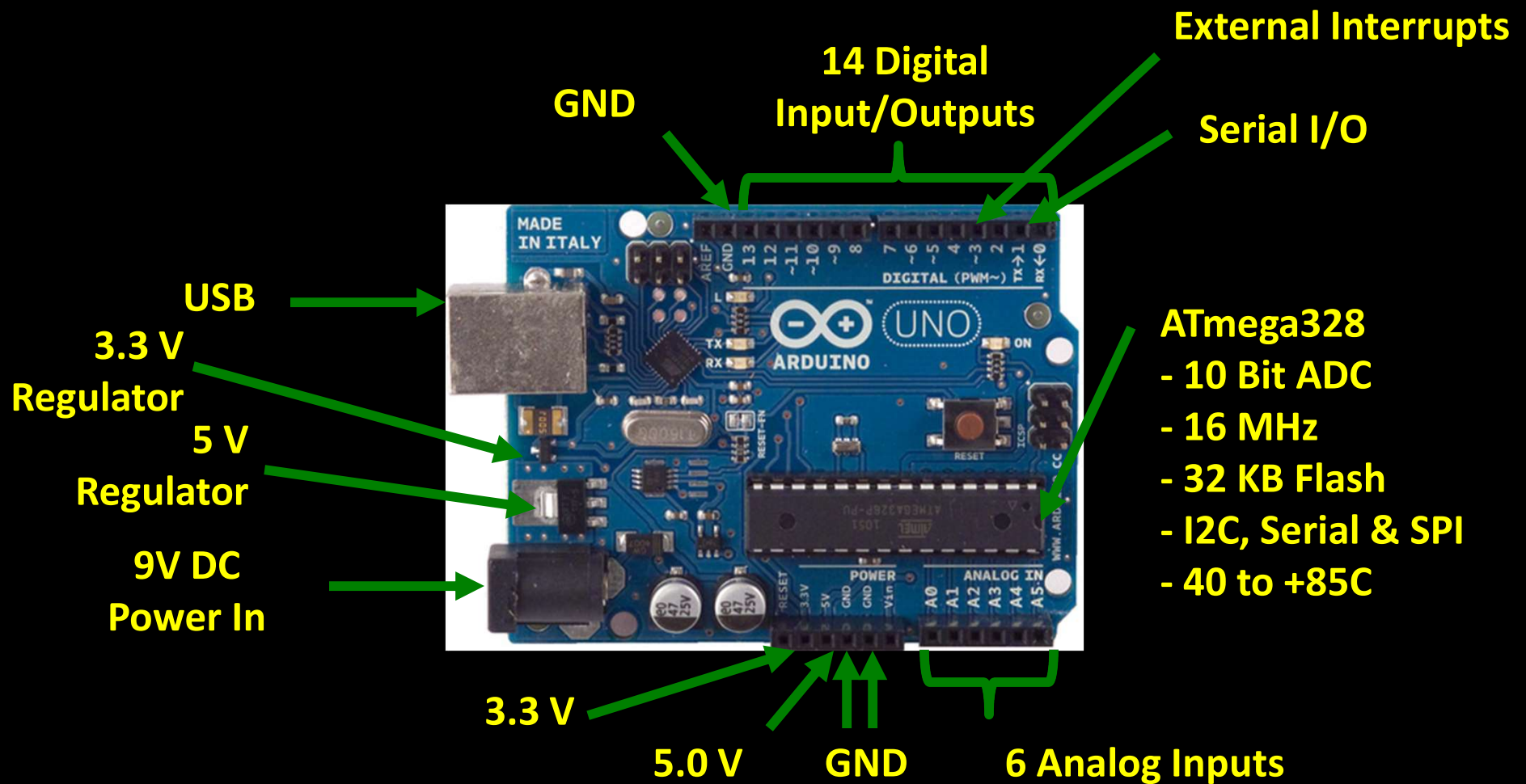


Arduino Overview:

Arduino(TM) UNO Rev3

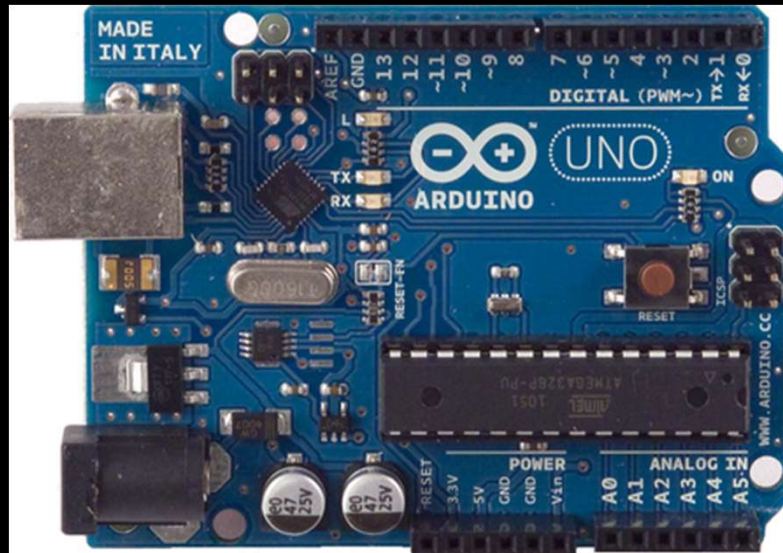


Arduino Overview:



Arduino Overview:

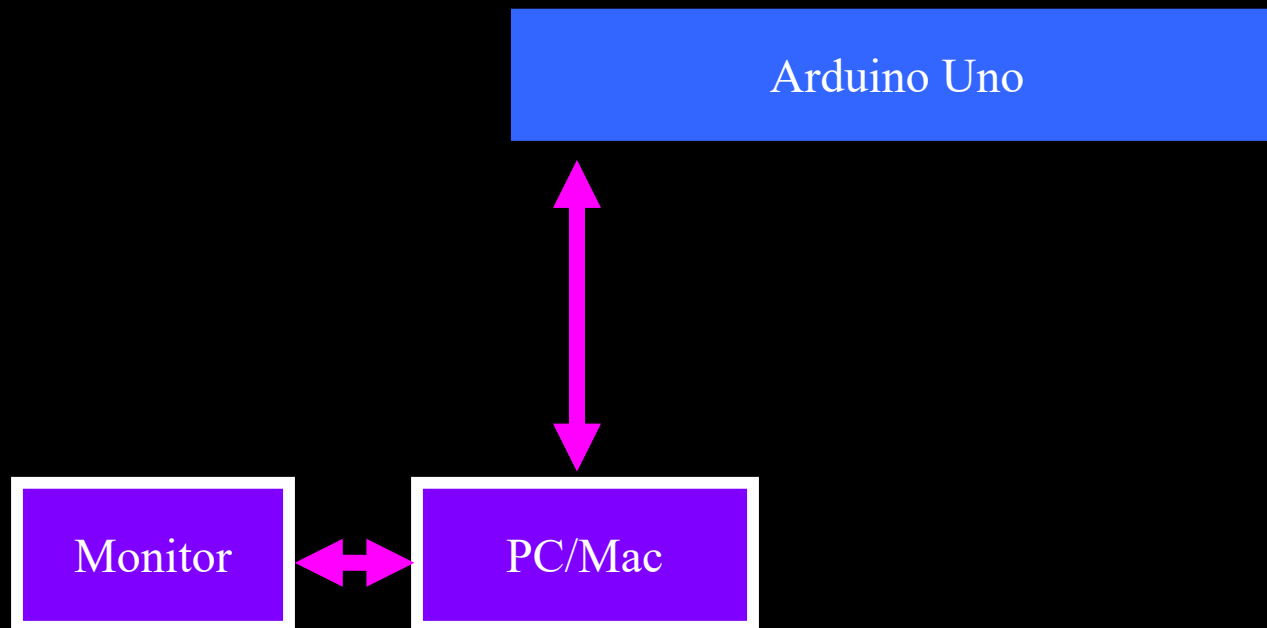
- Let's take it for a drive...



Arduino Overview:

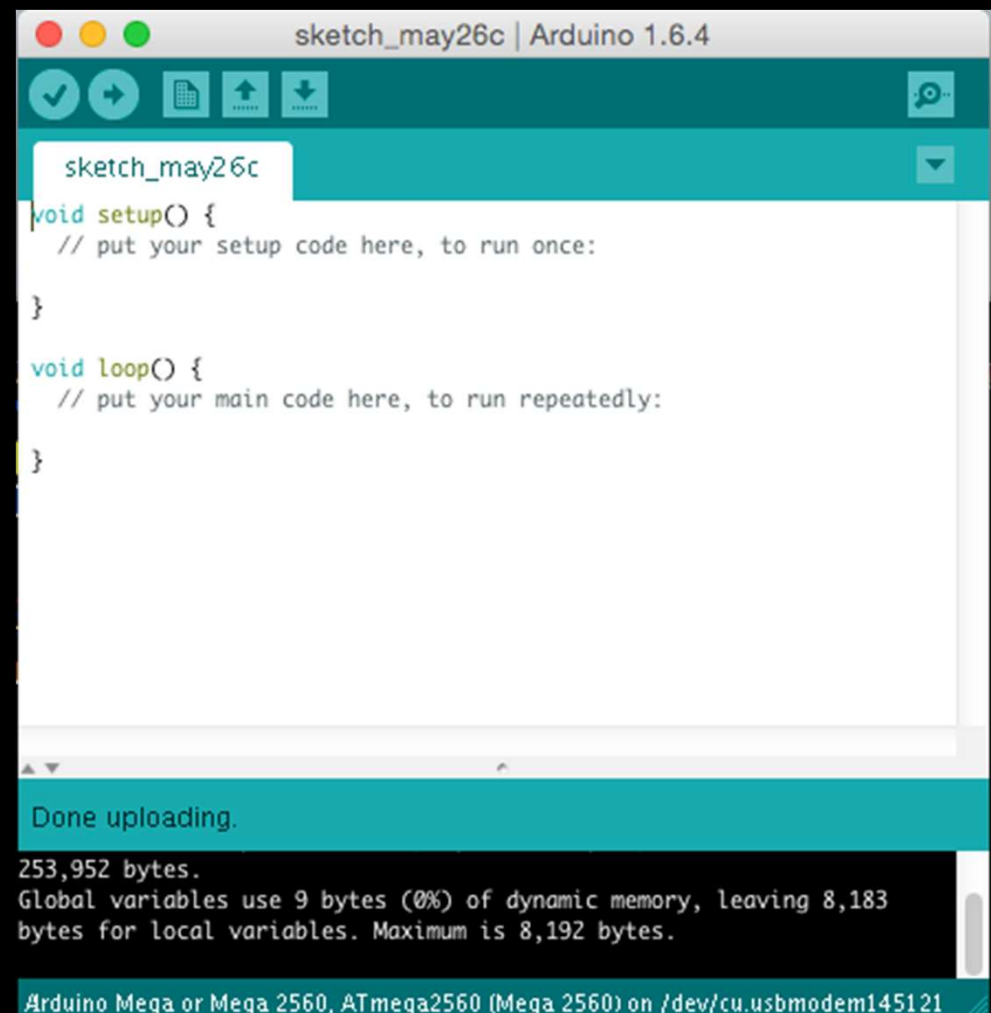
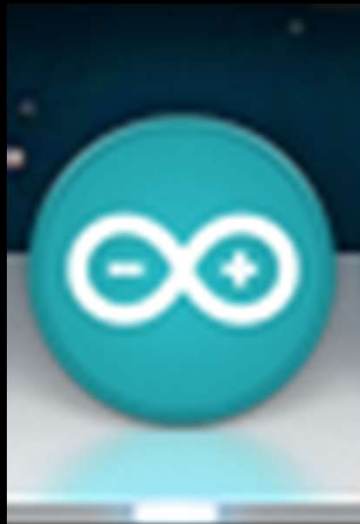
Arduino Uno

Arduino Overview:



Arduino Overview:

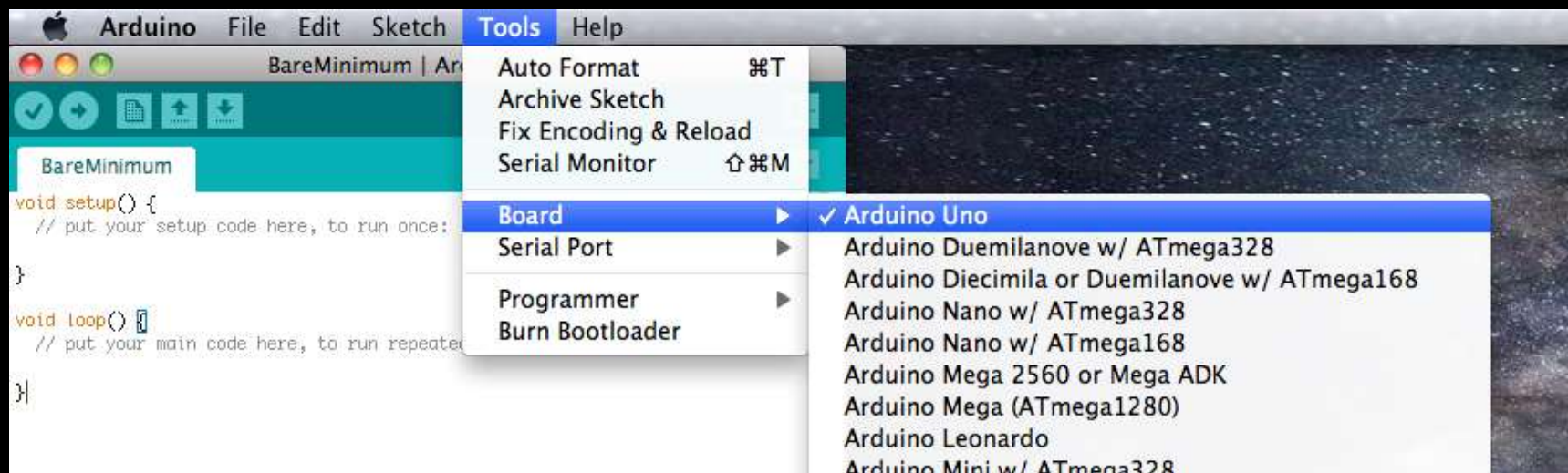
- Launch the Arduino Software
- A new Sketch opens



Arduino Overview:

Select the right board =

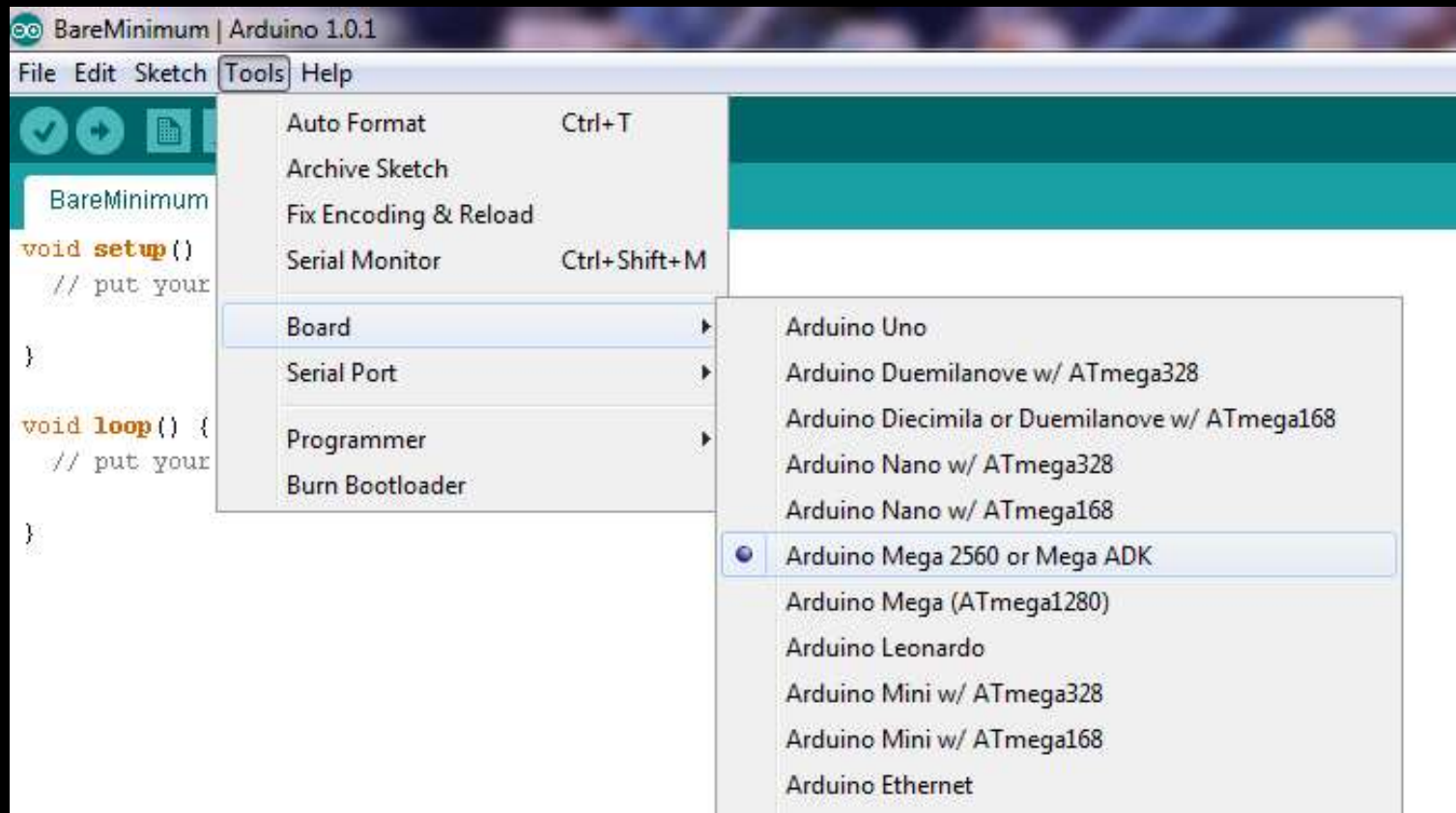
Tools > Board > Arduino Uno



Arduino Overview:

Select the right board =

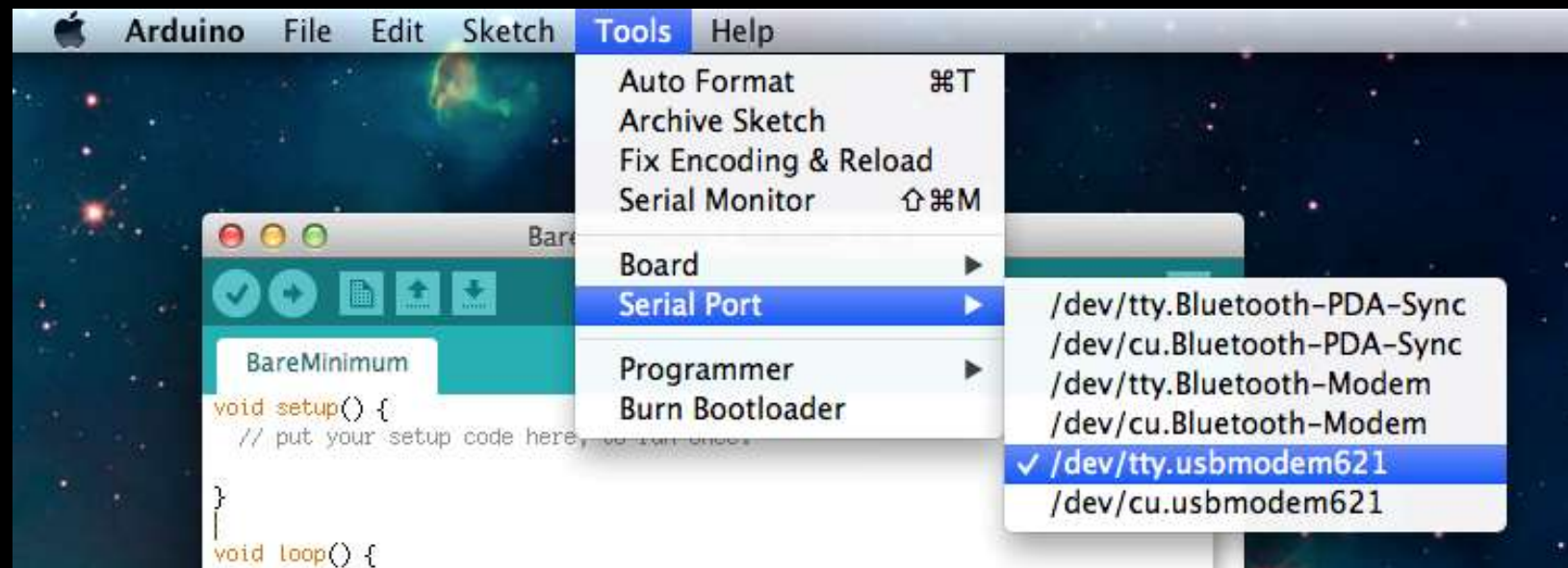
Tools > Board > Arduino Uno





Arduino Overview

- Select a serial port
 - For Mac use **Tools > Serial Port > /dev/tty.usbmodemxxx**
 - Note: the 'xxx' and 'xx' above can be any number – it does not matter which number you choose as long as one is selected

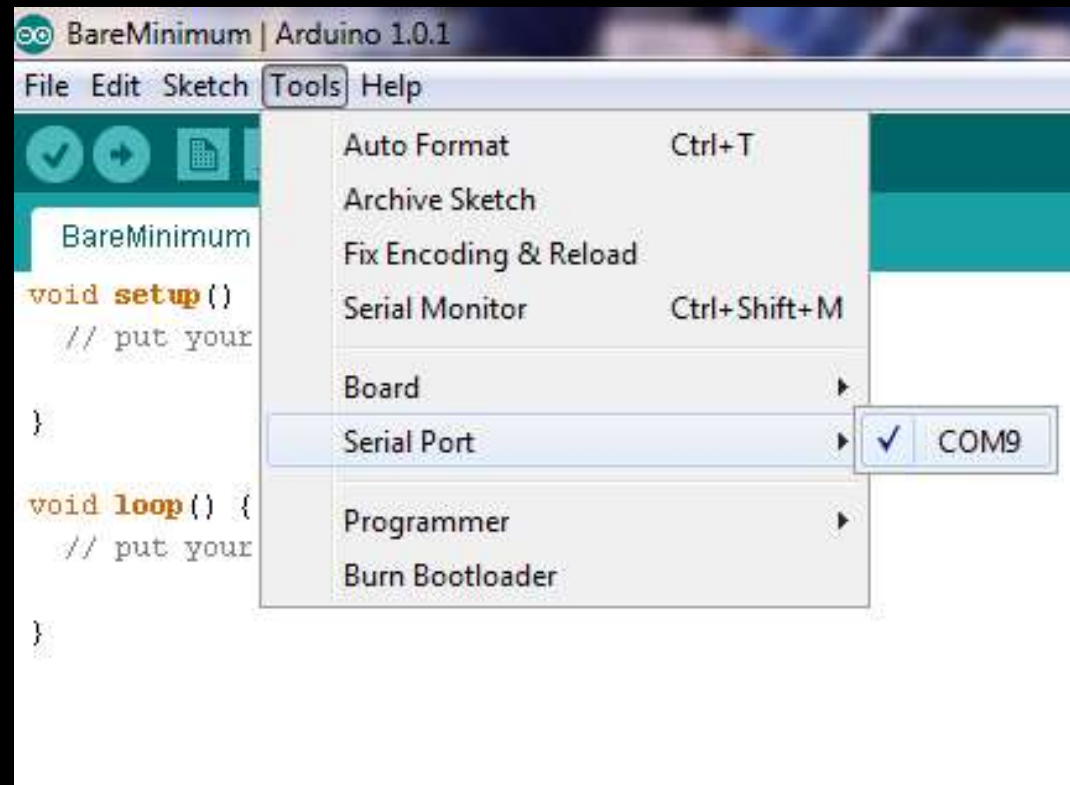




Arduino Overview:

- Select a serial port
 - For PC use **Tools > Serial Port > COMxx**

Note: the 'xxx' and 'xx' above can be any number – it does not matter which number you choose as long as one is selected

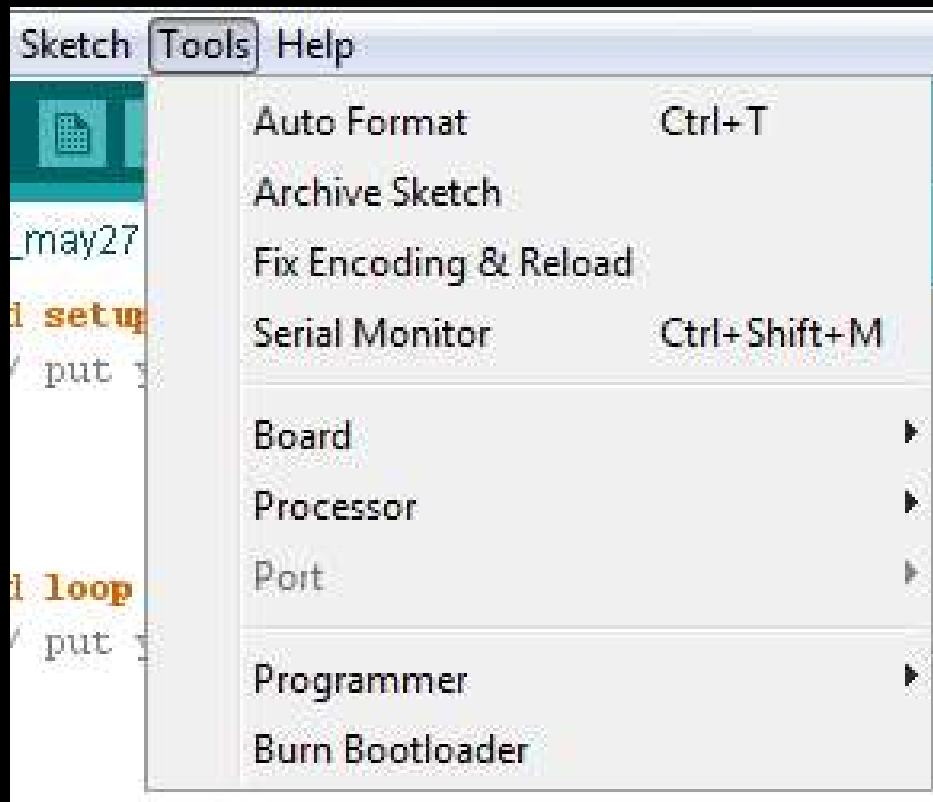


Arduino Overview:

Port is big source of frustration for Windows users



- The dreaded “**grayed out**” port



When it happens...

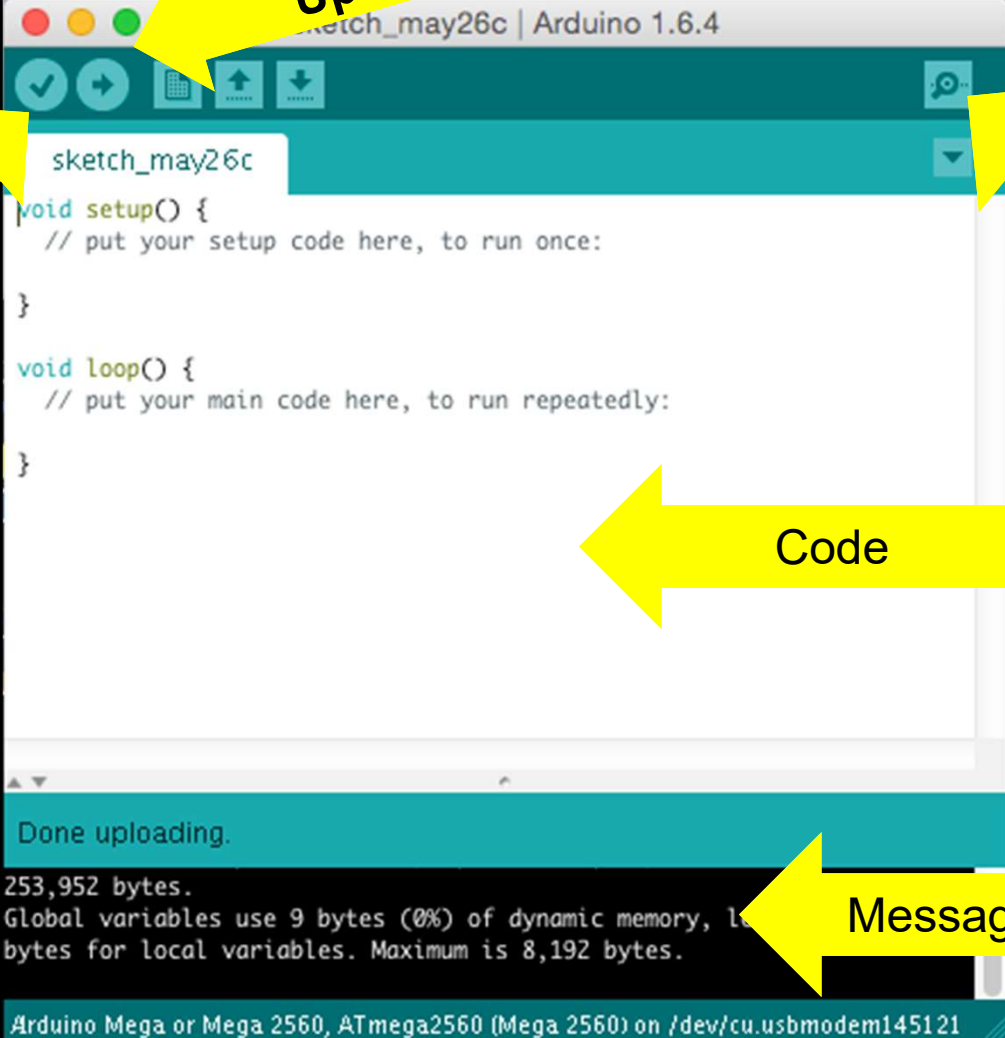
- Unplug Arduino from laptop
- Close Arduino software
- Plug Arduino back into laptop
- Restart Arduino software

Arduino Overview:

Review the Sketch



Compile



```
sketch_may26c | Arduino 1.6.4  
  
void setup() {  
  // put your setup code here, to run once:  
}  
  
void loop() {  
  // put your main code here, to run repeatedly:  
}
```

Done uploading.
253,952 bytes.
Global variables use 9 bytes (0%) of dynamic memory, 1 bytes for local variables. Maximum is 8,192 bytes.
Arduino Mega or Mega 2560, ATmega2560 (Mega 2560) on /dev/cu.usbmodem145121

Upload

Code

Serial Monitor

Message Box



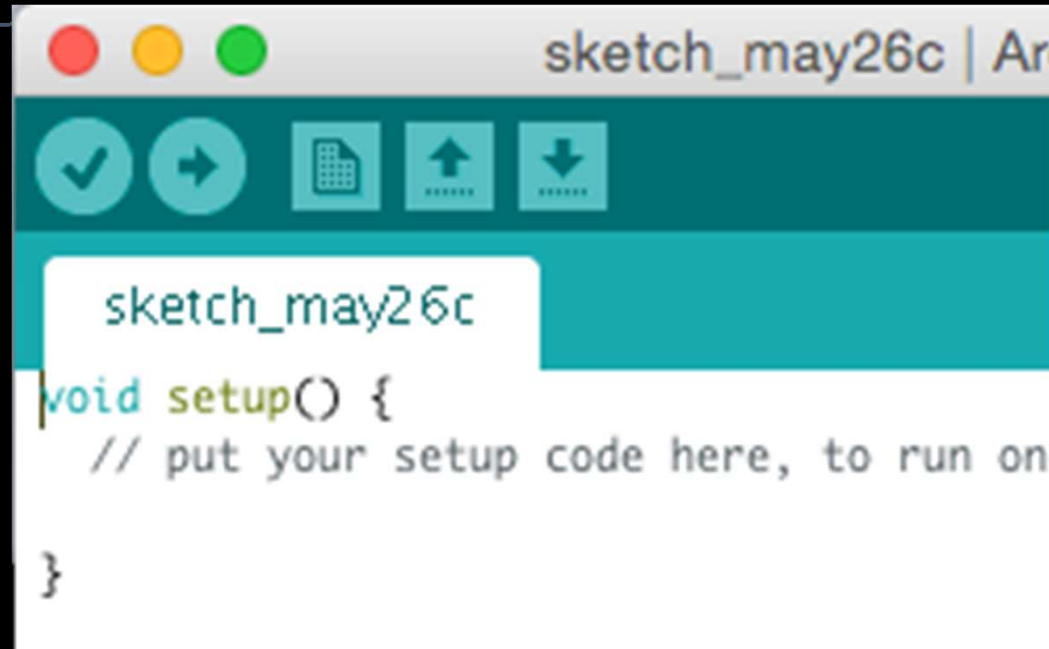
Arduino Overview:

There are three main sections of code in an Arduino sketch:

- Definitions**
- Void Setup**
- Void Loop**

Arduino Overview:

Definitions

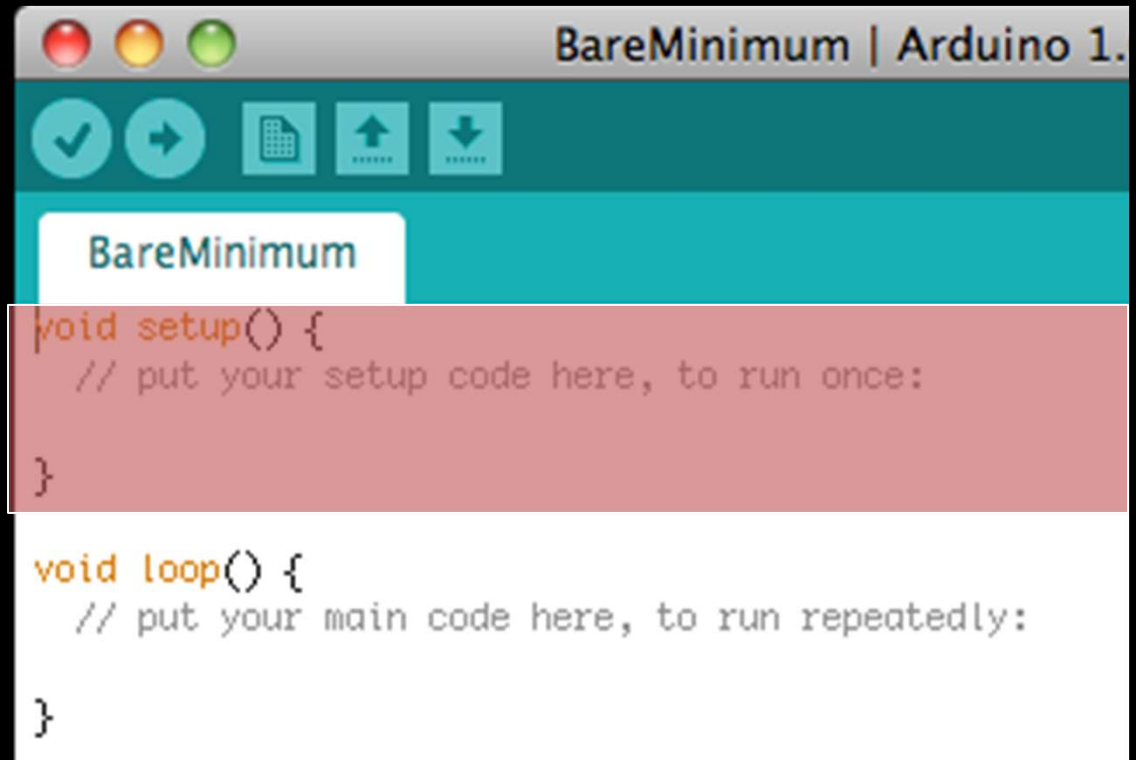


- Definitions are declared **prior to void setup** and can include pin definitions, libraries to include in the sketch, functions, and global variables
- Most programs declare something, but this is not required. Examples later on...

Arduino Overview:

void setup

- **void setup** is the first code block in the sketch
- It is run only once
- Used for setup of pin modes, communication initialization, and any code we only want to run one time



```
BareMinimum | Arduino 1.

void setup() {
  // put your setup code here, to run once:
}

void loop() {
  // put your main code here, to run repeatedly:
}
```

Arduino Overview:

void loop



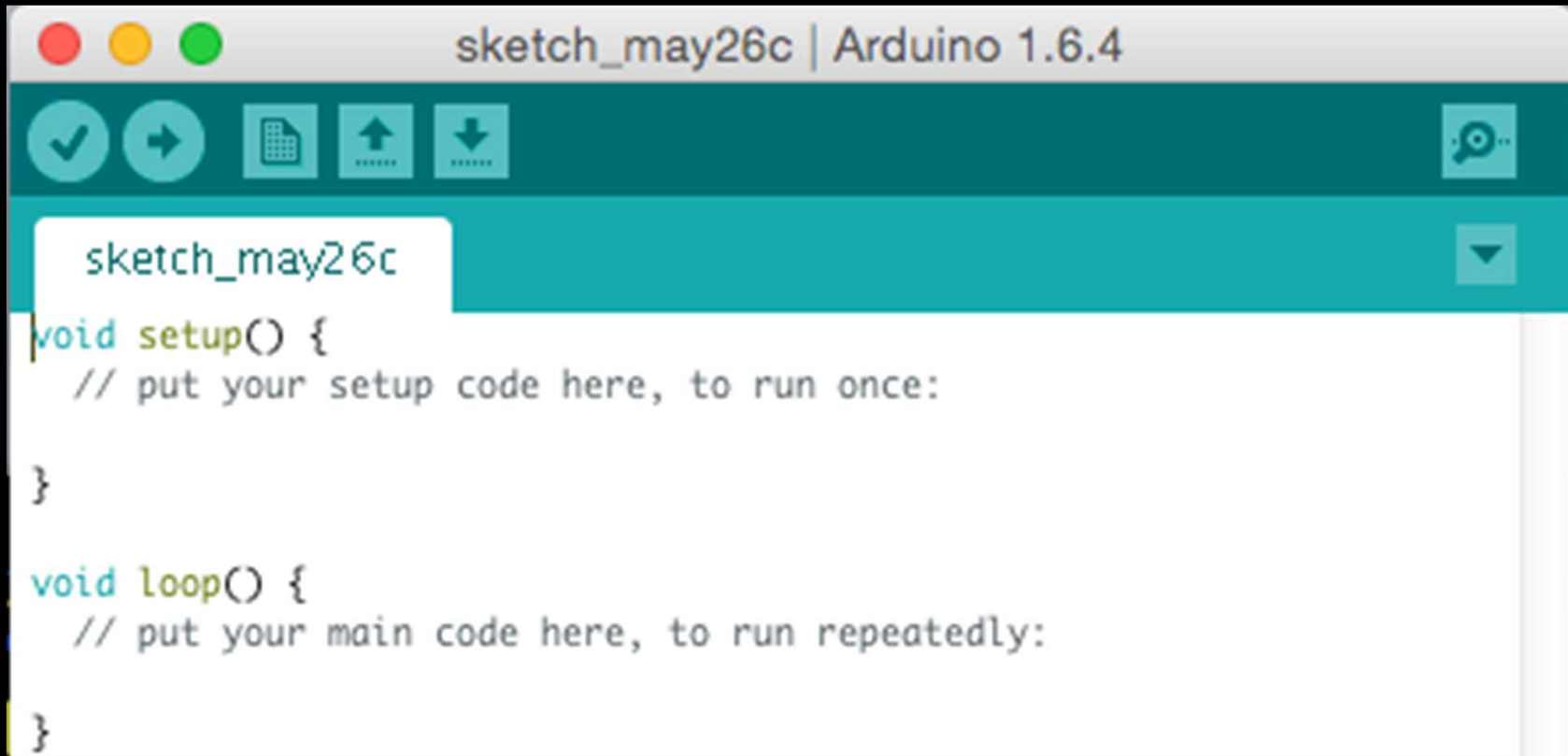
```
BareMinimum | Arduino 1.

void setup() {
  // put your setup code here, to run once:
}

void loop() {
  // put your main code here, to run repeatedly:
}
```

- **void loop** is the second code block in the Arduino sketch and it continuously repeats itself
- For code that needs to repeat such as sampling a sensor every couple of seconds
- Where the primary tasks of the code are carried out

Arduino Overview:



The screenshot shows the Arduino IDE window titled "sketch_may26c | Arduino 1.6.4". The interface includes a toolbar with icons for checking, running, saving, and uploading. The main editor area displays the following code:

```
sketch_may26c
void setup() {
  // put your setup code here, to run once:
}

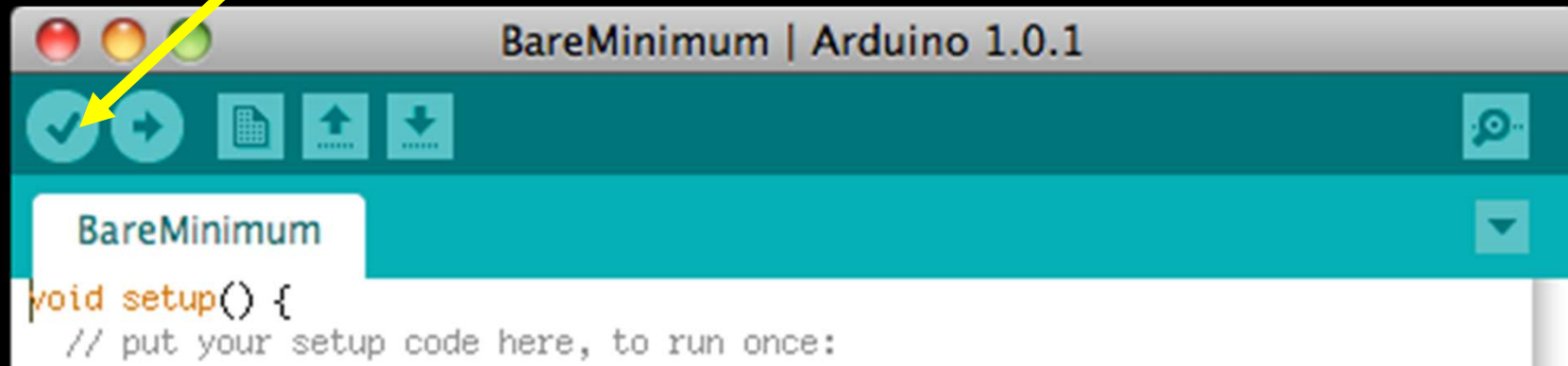
void loop() {
  // put your main code here, to run repeatedly:
}
```

- **Even though this Sketch is not doing anything, it has all the necessary ingredients to be compiled and uploaded**

Arduino Overview:

1. Compile code and check for messages

#1



Should see this at the bottom...



Arduino Overview:

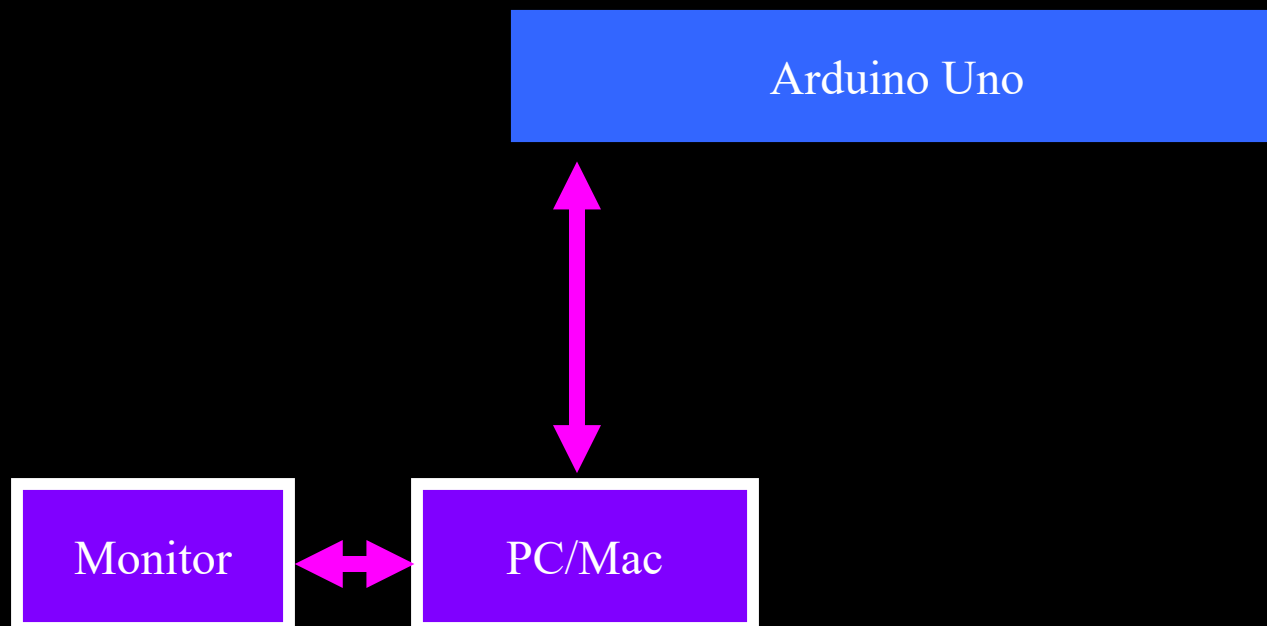
- **What is Compiling?**
 - It checks your code for syntax errors and returns error messages
 - Converts human-readable code into machine language (zeroes and ones)
 - When you tell the Arduino to upload, it first compiles then uploads (programs) your code **(communicating with laptop and Arduino)**



Arduino Driving Lessons

- A. Download Arduino IDE**
- B. Arduino Overview**
- C. Arduino Communication**
- D. Blink an Led, Change the World**

Arduino Overview:



Arduino Communication:

Arduino uses **serial communication** to communicate with your laptop.

Serial communication is a widely used interface* for transmitting (Tx) and receiving (Rx) binary data and requires a few easy functions to get it started with Arduino.

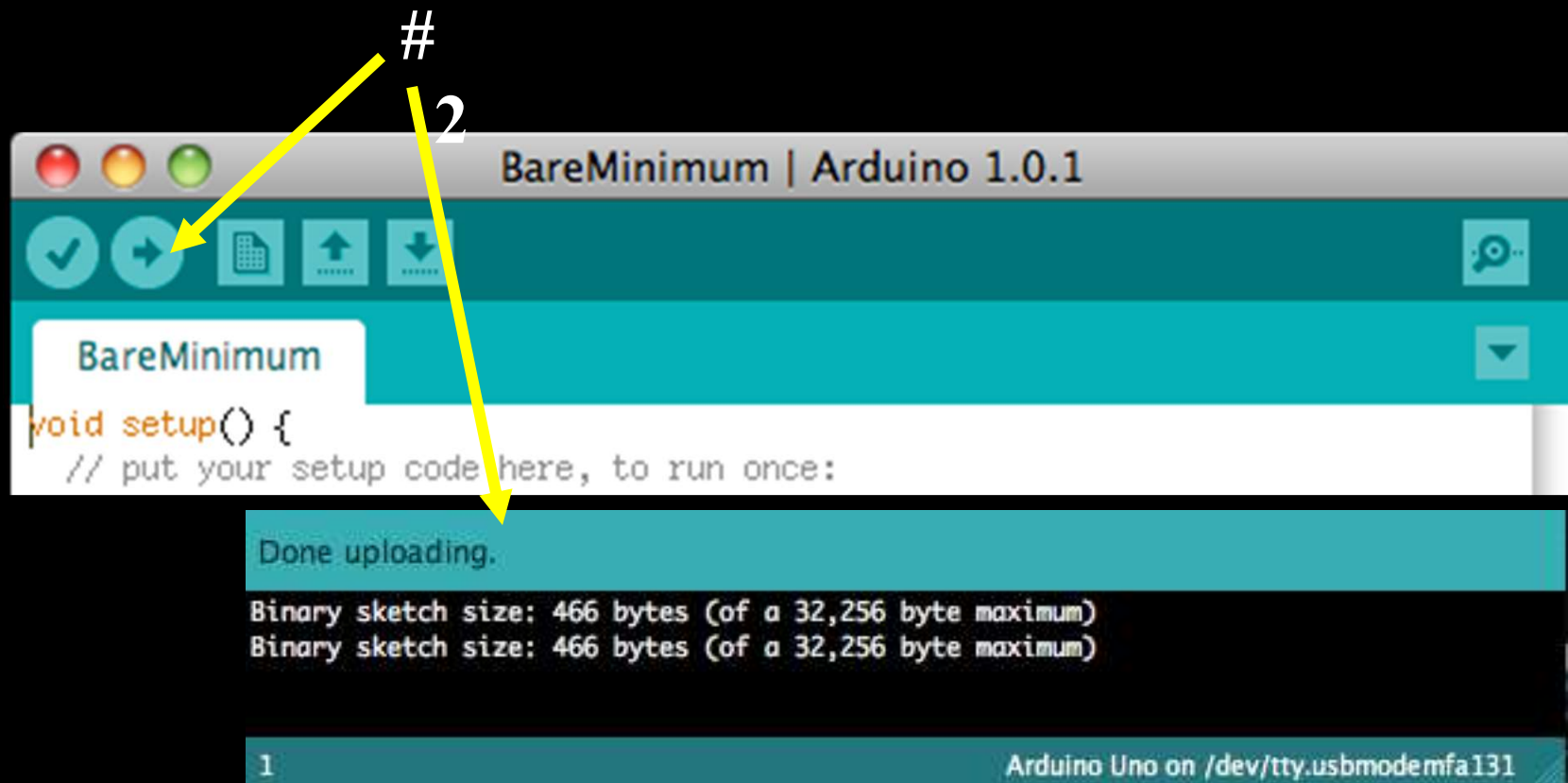


* UART (common set of communication rules)

Arduino Communication:

2. Upload code to Arduino

If successfully uploaded, you will know that your PC/MAC can communicate with your Arduino



Arduino Communication:



Space Minor
UNIVERSITY OF COLORADO BOULDER

What is code?

Arduino Communication:



Space Minor
UNIVERSITY OF COLORADO BOULDER

What is code?

- It is a language to talk with your computer
- Programming languages are like foreign languages
- We say “Hello,” Arduino says
Serial.begin(9600);
Serial.print(“Hello”);
- Arduino language is based on C/C++

Code Checklist:



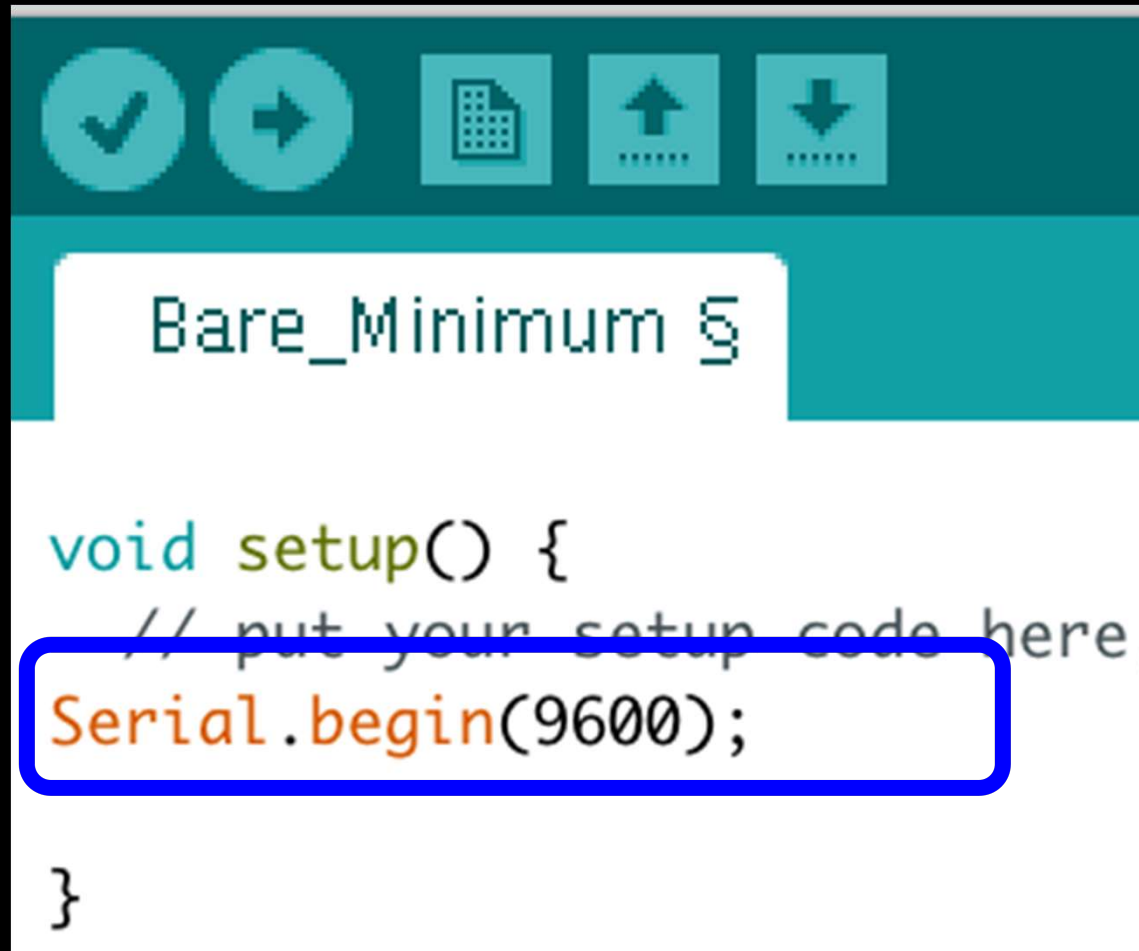
Space Minor

UNIVERSITY OF COLORADO BOULDER

[illegible]

Arduino Communication:

- Modify the sketch to add the following to the **void setup()**

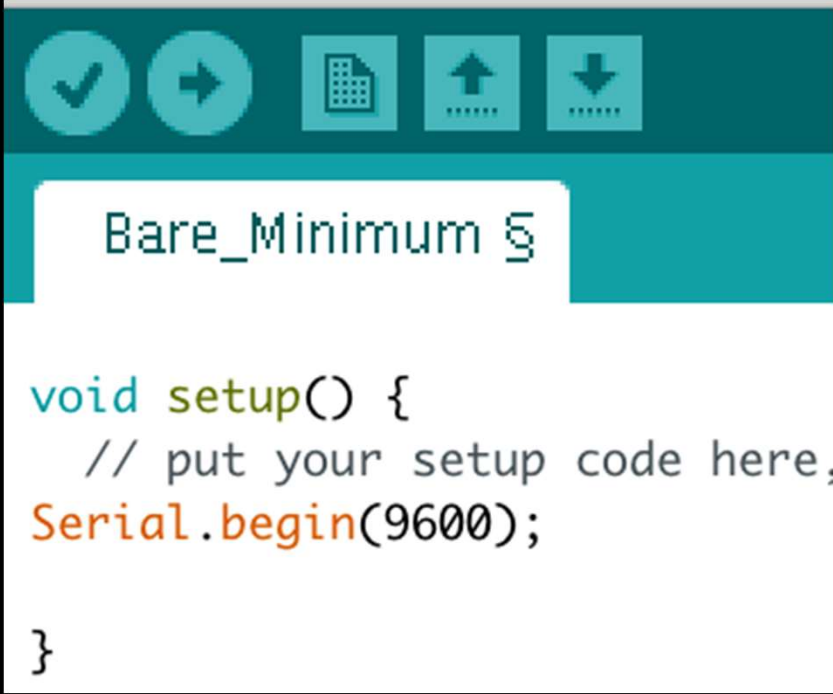


```
Bare_Minimum §  
  
void setup() {  
  // put your setup code here,  
  Serial.begin(9600);  
}
```

Arduino Communication:

Serial.begin()

- **Serial.begin()** needs us to specify a communication rate (baud rate)
- We use 9600 bits per second, so put 9600 in the parentheses
- **Serial.begin()** is in **setup** because this rate needs to be set only once



```
void setup() {  
  // put your setup code here,  
  Serial.begin(9600);  
}
```

Arduino Communication:

Modify your sketch to include the following:

```
void loop() {  
    // put your main code here, to run repeatedly:  
    Serial.println("hello");  
}
```

- **Serial.print()** will just print to the monitor
- **Serial.println()** will print to the monitor and then go to the next line (essentially pushes 'return')

Arduino Communication:

1. Compile code and check for messages
2. Upload code to Arduino (will check communication with Arduino too)



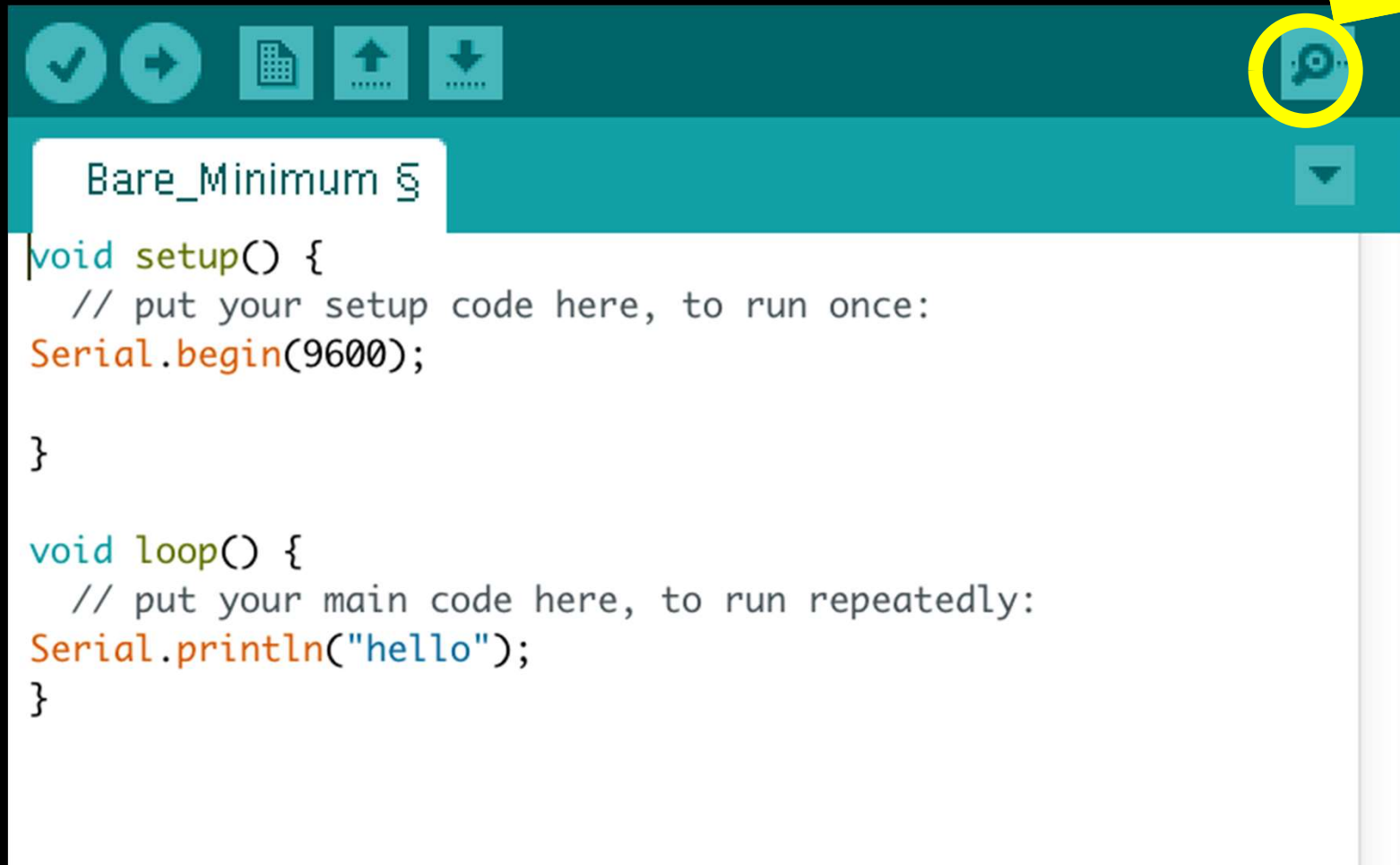
Arduino Communication:



Space Minor
UNIVERSITY OF COLORADO B

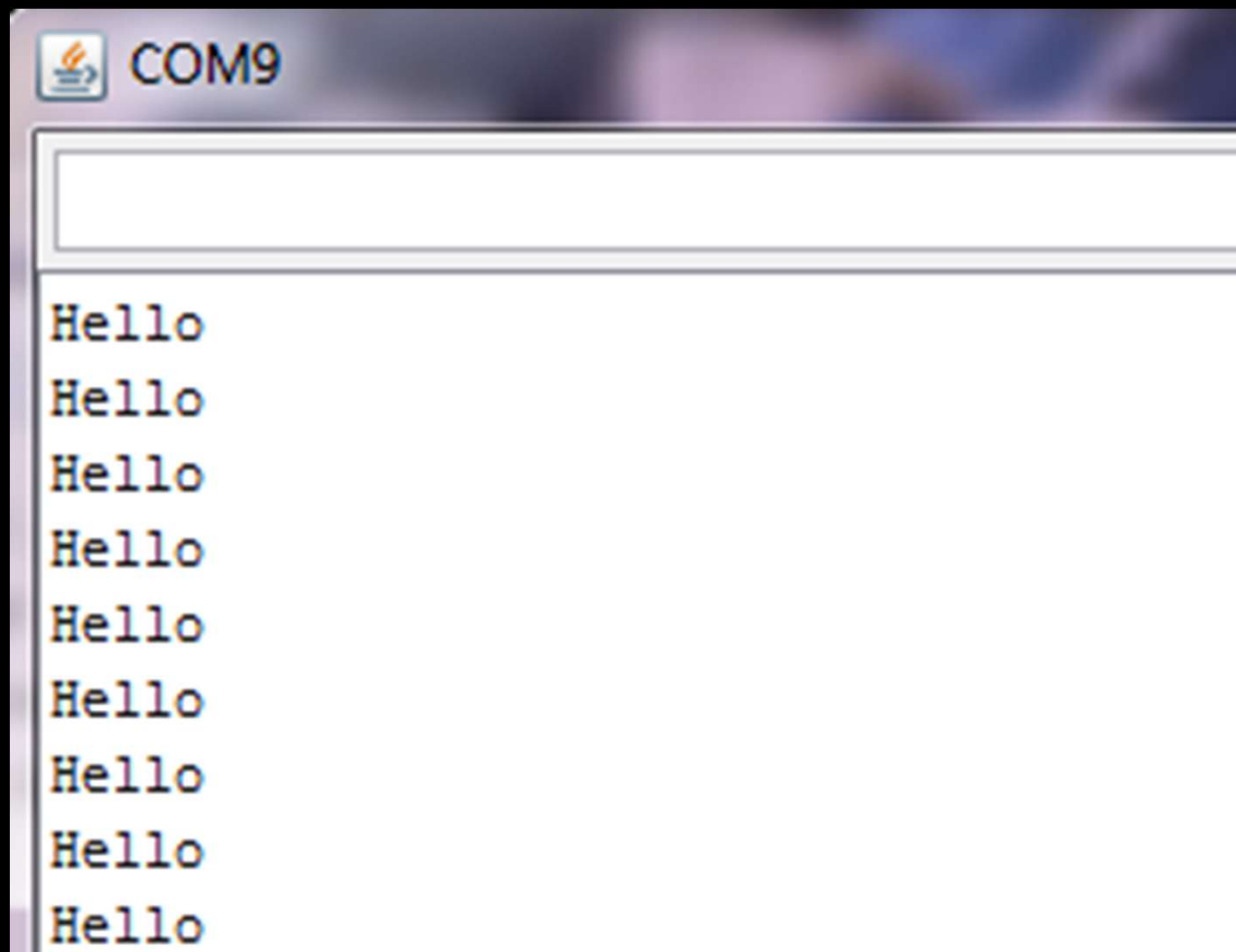
Serial Monitor

- To open the serial monitor, click here



Arduino Communication:

- You should see this on your serial monitor



Arduino Communication:

- Congratulations!
- You have now successfully programmed your Arduino – You are a computer programmer



A Brief History Break:

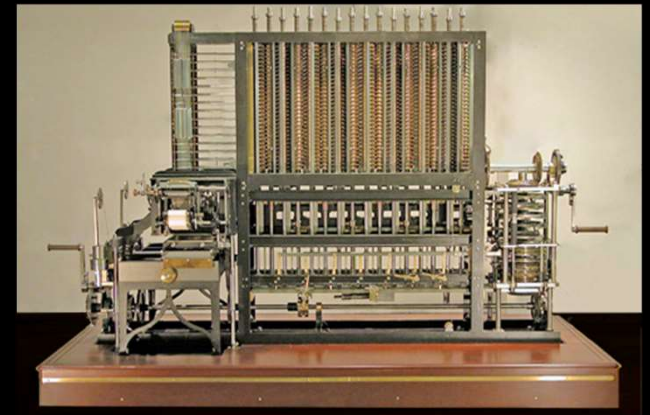
Ada Lovelace – Mathematician

First computer programmer in 1842

“Analytical Engine”

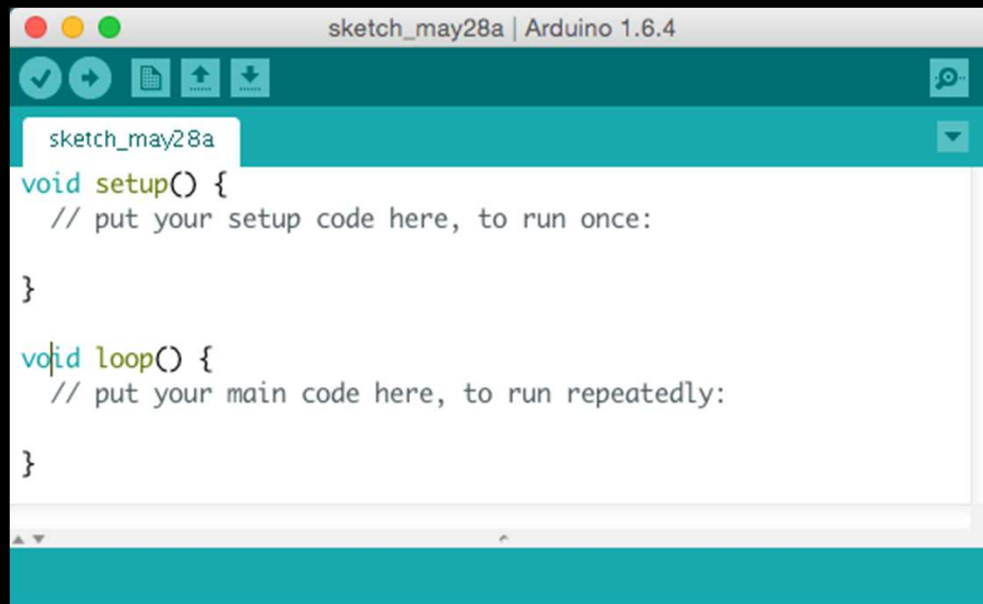
Saw that it was more than number crunching, but was a tool capable of great problem solving

Published most elaborate and complete programs



Arduino Preferences:

Other features about Arduino IDE



```
sketch_may28a
void setup() {
  // put your setup code here, to run once:
}

void loop() {
  // put your main code here, to run repeatedly:
}
```

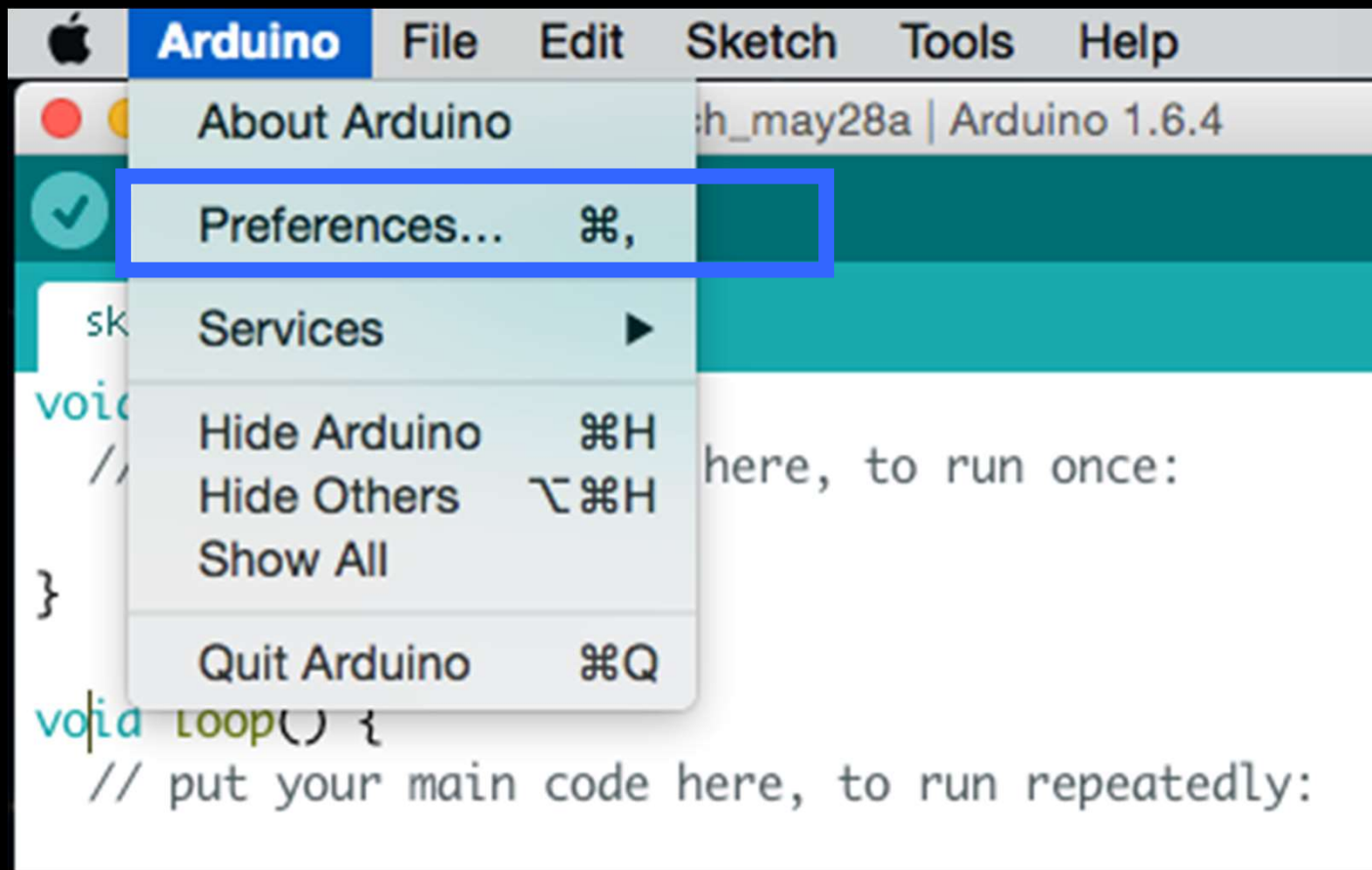
Line Number of
cursor in code

6

6 Arduino Mega or Mega 2560, ATmega2560 (Mega 2560) on /dev/cu.usbmodem145121

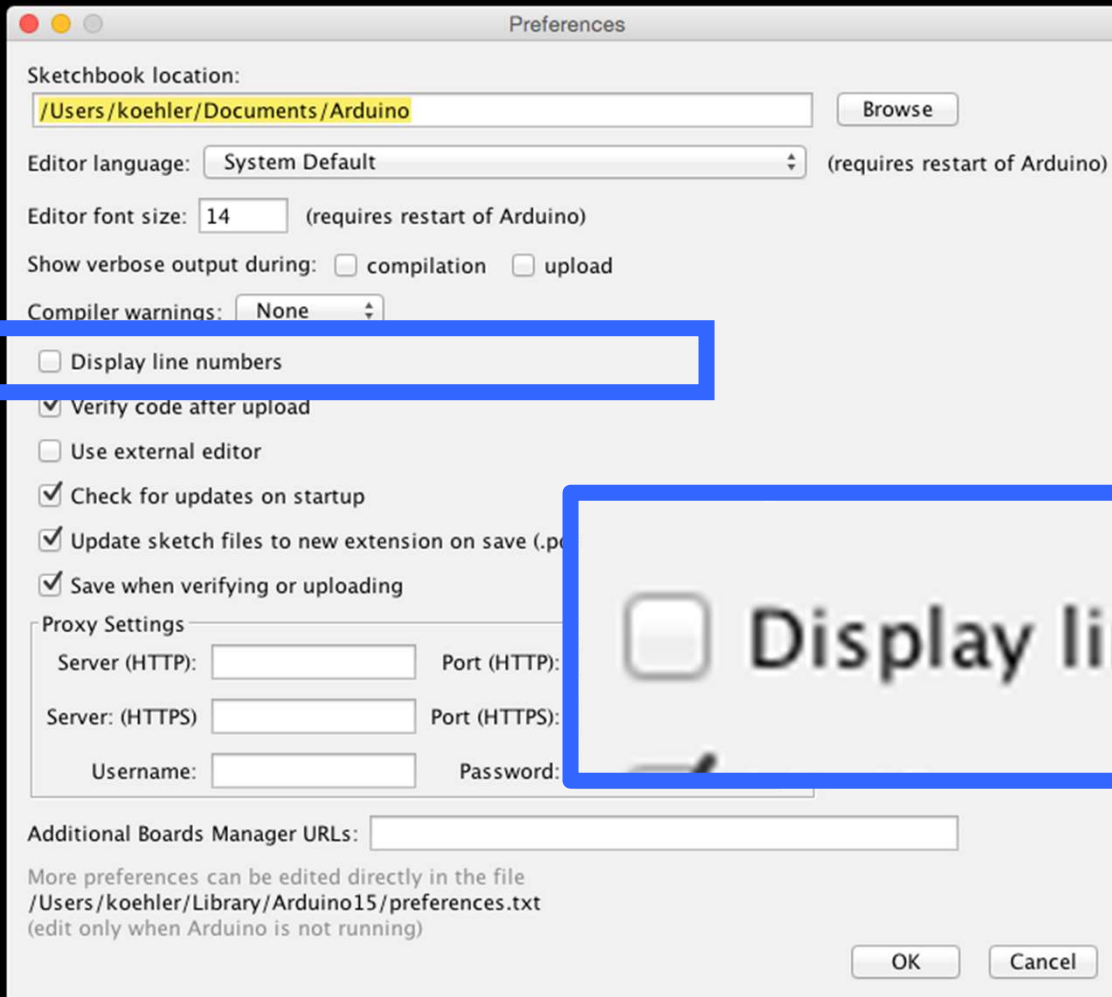
Arduino Preferences:

Other features about Arduino IDE



Arduino Preferences:

Other features about Arduino IDE



Preferences

Sketchbook location:

Editor language: (requires restart of Arduino)

Editor font size: (requires restart of Arduino)

Show verbose output during: ☐ compilation ☐ upload

Compiler warnings:

☐ Display line numbers

☒ Verify code after upload

☐ Use external editor

☒ Check for updates on startup

☒ Update sketch files to new extension on save (.p...)

☒ Save when verifying or uploading

Proxy Settings

Server (HTTP): Port (HTTP):

Server (HTTPS): Port (HTTPS):

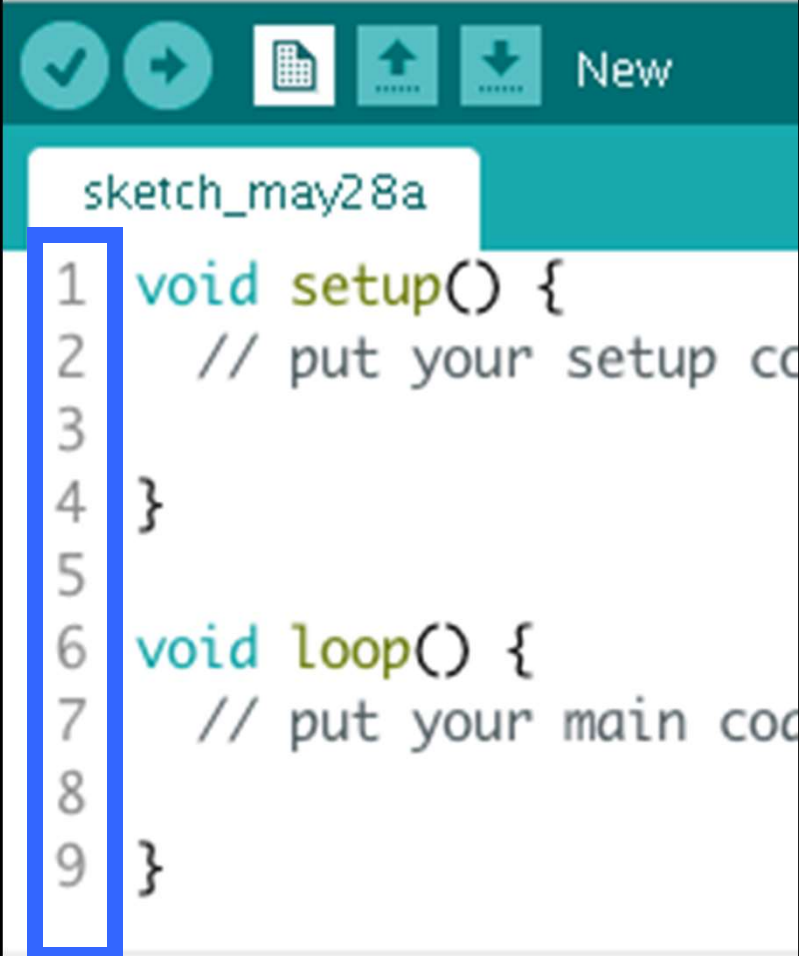
Username: Password:

Additional Boards Manager URLs:

More preferences can be edited directly in the file
/Users/koehler/Library/Arduino15/preferences.txt
(edit only when Arduino is not running)

Arduino Preferences:

Other features about Arduino IDE

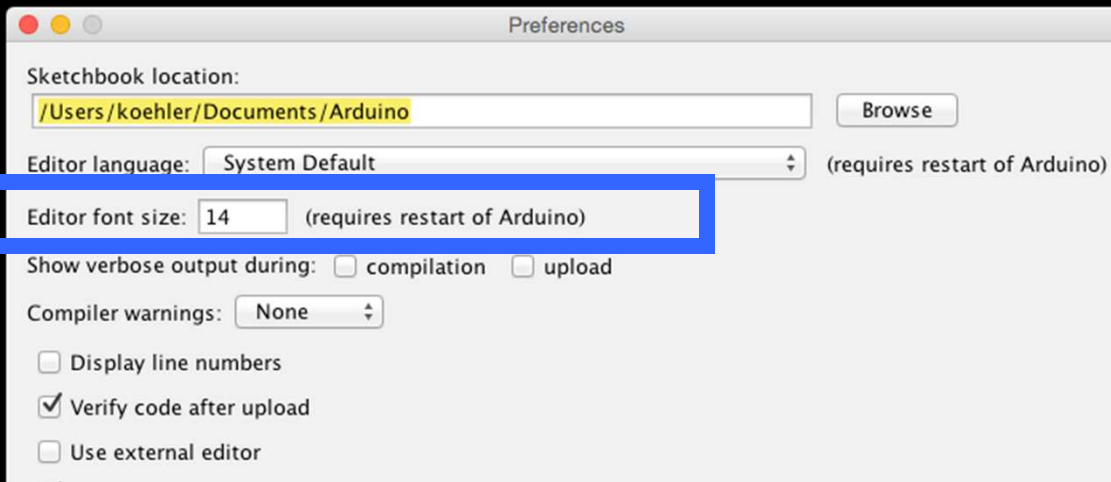


```
sketch_may28a

1 void setup() {
2   // put your setup code here, to initialize vars:
3
4 }
5
6 void loop() {
7   // put your main code here, to run repeatedly:
8
9 }
```

Arduino Preferences:

Other features about Arduino IDE



Preferences

Sketchbook location:

Editor language: (requires restart of Arduino)

Editor font size: (requires restart of Arduino)

Show verbose output during: ☐ compilation ☐ upload

Compiler warnings:

☐ Display line numbers

☒ Verify code after upload

☐ Use external editor

Editor font size: (requires restart of Arduino)

Username: Password:

Additional Boards Manager URLs:

More preferences can be edited directly in the file
/Users/koehler/Library/Arduino15/preferences.txt
(edit only when Arduino is not running)

Arduino Communication:

Commenting

- Arduino ignores comments but humans read them
- Words become light gray if they commented out
- Put **//** in front of a line to comment out whole line
- To comment out an entire section, put **/*** at the beginning and ***/** at the end

```
//you can type anything you want here!  
this is NOT a comment!! //uh oh!  
/*  
I can type whatever I want here.  
Notice how it's gray?  
*/
```

Arduino Communication:

Commenting – MOST IMPORTANT THING!!

It makes your code readable, provides context, helps draft out what you want to next.

- Click to the top of the sketch hit enter to create a new line above `void setup()`
- Try out your own comment. Insert your name at the top of the sketch. Try both methods.



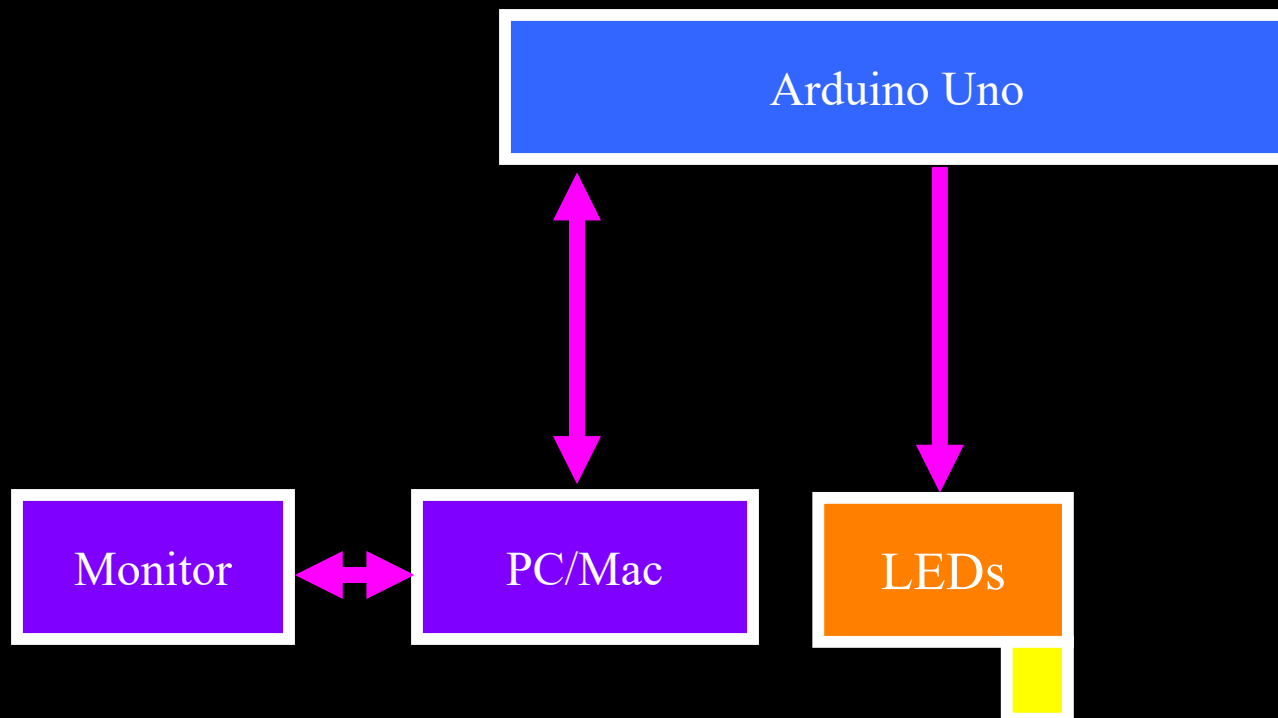
```
BareMinimum $
//RockOn Workshop 2013
/*RockOn Workshop 2013*/
void setup() {
  // put your setup code here, to run once:
}
```



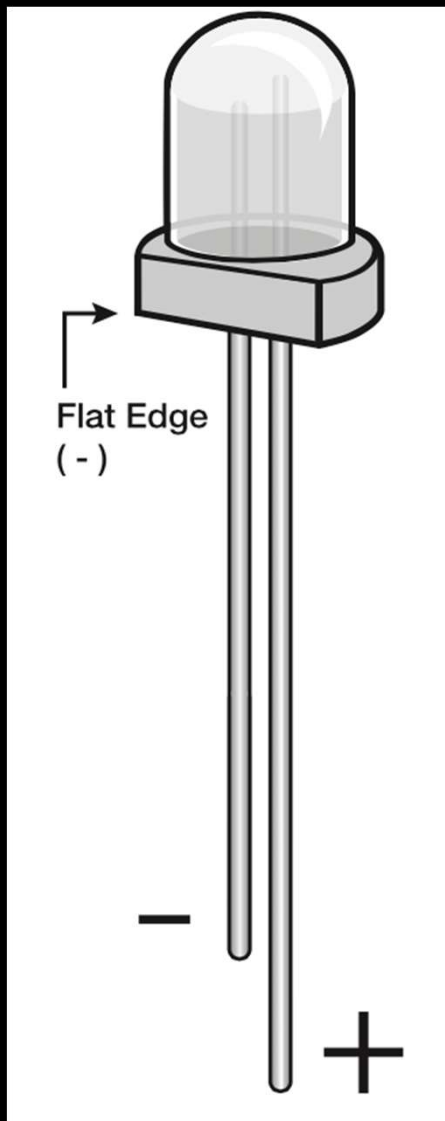
Arduino Driving Lessons

- A. Download Arduino IDE
- B. Arduino Overview
- C. Arduino Communication
- D. Blink an Led, Change the World

Blink an LED:

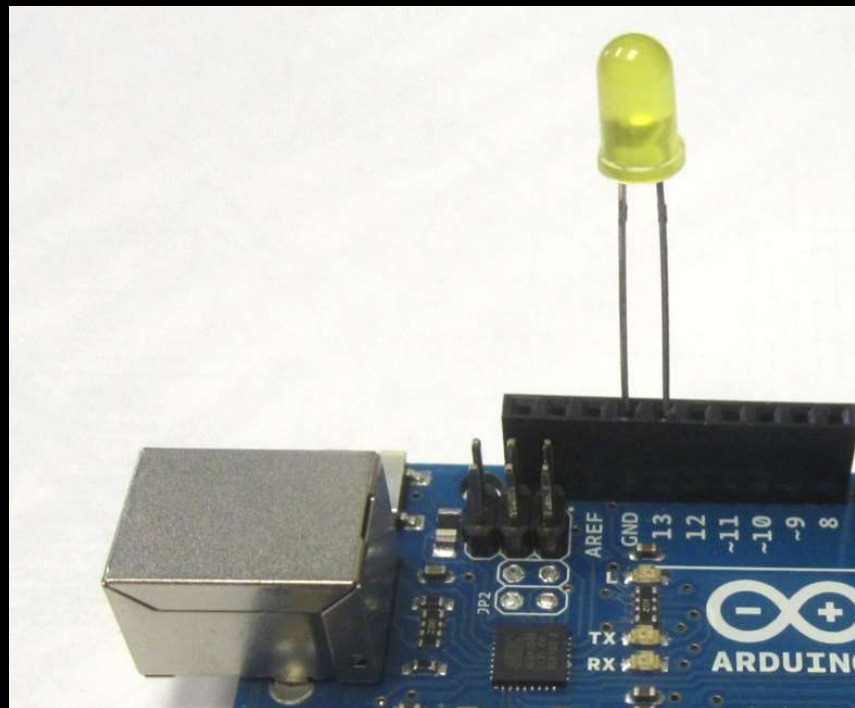


Blink an LED:



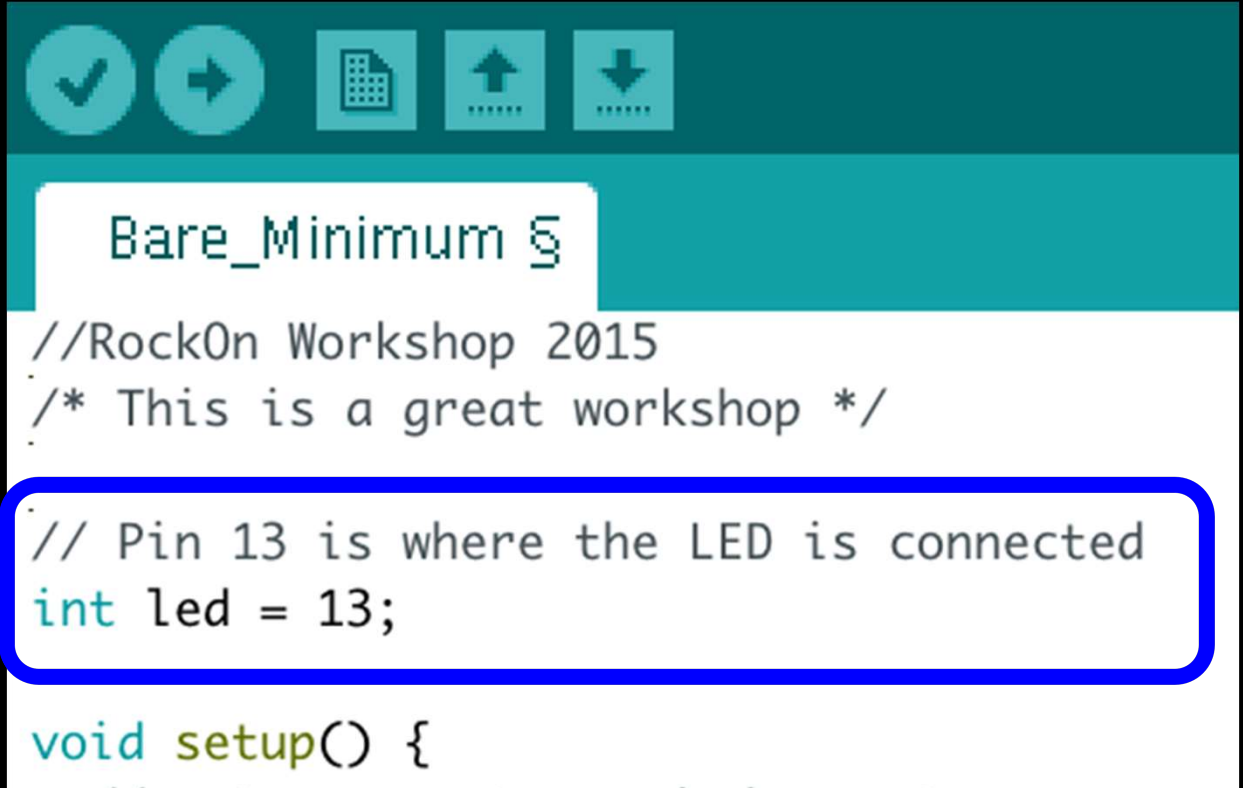
- Connect an LED (+ lead) to pin 13 and (- lead) to GND

- Negative lead is usually the shorter lead



Blink an LED:

- Add the following to the **definitions** area of your sketch - *above void setup()*
- Add some comments too



```
Bare_Minimum §  
//RockOn Workshop 2015  
/* This is a great workshop */  
  
// Pin 13 is where the LED is connected  
int led = 13;  
  
void setup() {
```

Blink an LED:

- Note that “**int**” turned blue → we are defining a data type
- Arduino knows variable “**led**” represents an **integer**
- Anytime “**led**” is used in the code, Arduino sees 13
 - Ex. $12 + \text{led} = 25$
- There are many more data types

```
// Pin 13 is where  
int led = 13;
```

Blink an LED:



Space Minor
UNIVERSITY OF COLORADO BOULDER

- Add the following to the your sketch in **void setup()**

```
void setup() {  
    // put your setup code here, to run once:  
    // initialize the digital pin as an output  
    pinMode(led, OUTPUT);  
    Serial.begin(9600);  
}
```

Blink an LED:

- **pinMode(pin#, mode)**

- “**pin#**” refers to a specific pin on the Arduino you are wanting to use (in our case pin 13 aka “led”)
- “**mode**” is either INPUT or OUTPUT
 - **OUTPUT** sets up the pin so it can give outputs
 - **INPUT** sets up the pin so it can receive inputs

```
// initialize the led  
pinMode(led, OUTPUT);  
// initialize serial
```

Pin 13 is now an output

Blink an LED:



Space Minor
UNIVERSITY OF COLORADO BOULDER

- Add the following to your sketch in **void loop()**

```
void loop() {  
    // put your main code here,  
    Serial.println("hello");  
    digitalWrite(led, HIGH);  
    delay(1000);  
    digitalWrite(led, LOW);  
    delay(1000);  
}
```

- **void loop () ...**
- Runs once void setup is finished
- Loops through the code within forever

Blink an LED:

digitalWrite(pin#, value)

- “**pin#**” is whichever pin you are writing to
- “**value**” can be either HIGH or LOW

- **HIGH** means the pin is at 5V – “on”
- **LOW** means the pin is at 0V – “off”

```
Serial.println("Hello");  
digitalWrite(led, HIGH);  
delay(1000);  
digitalWrite(led, LOW);  
delay(1000);  
}
```

Blink an LED:

delay(time)

- Arduino will wait a specific amount of time (in milliseconds) before going to the next line of code

```
Serial.println("Hello");  
digitalWrite(led, HIGH);  
delay(1000);  
digitalWrite(led, LOW);  
delay(1000);  
}
```

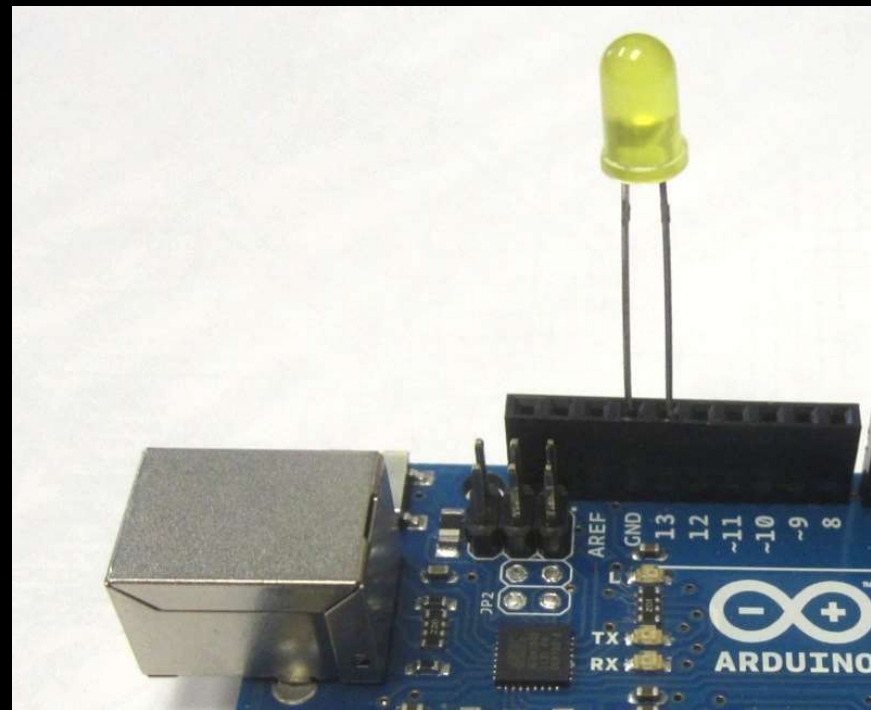
Blink an LED:

1. Compile code and check for messages
2. Upload code to Arduino



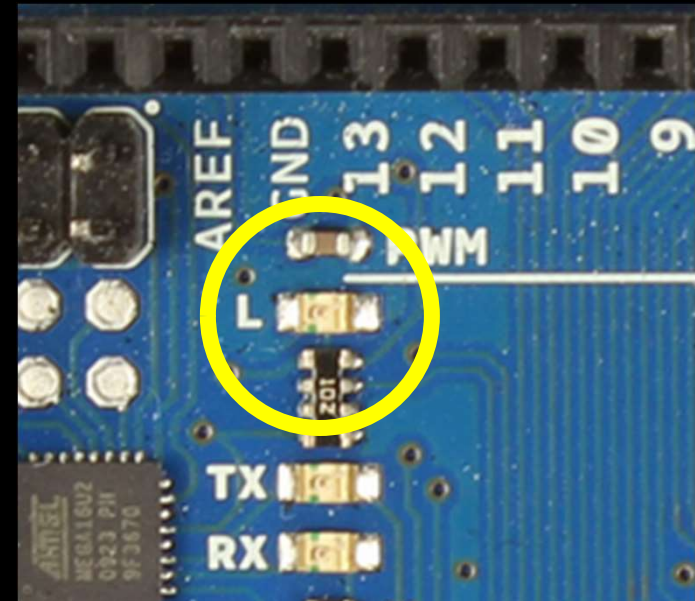
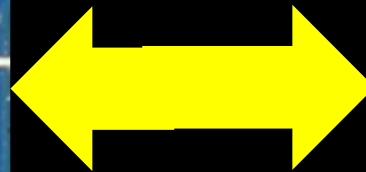
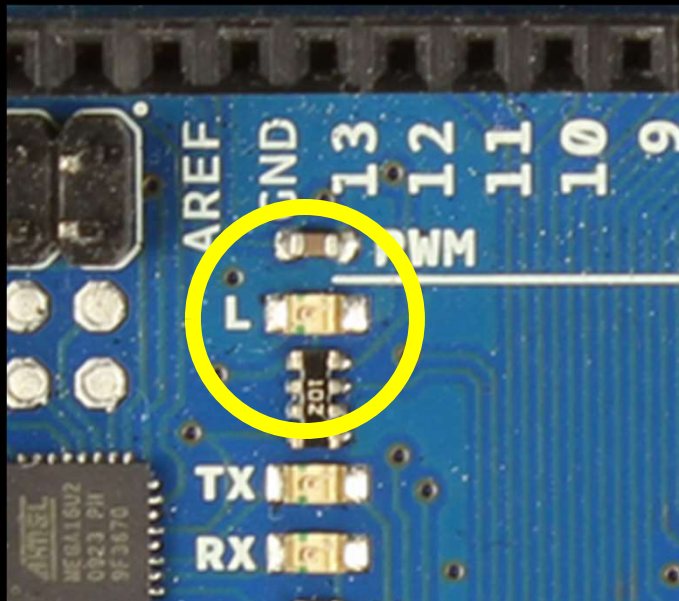
Blink an LED:

- Does LED blink?
- Change the delay in the sketch and try again
- Do you see a change?



Blink an LED:

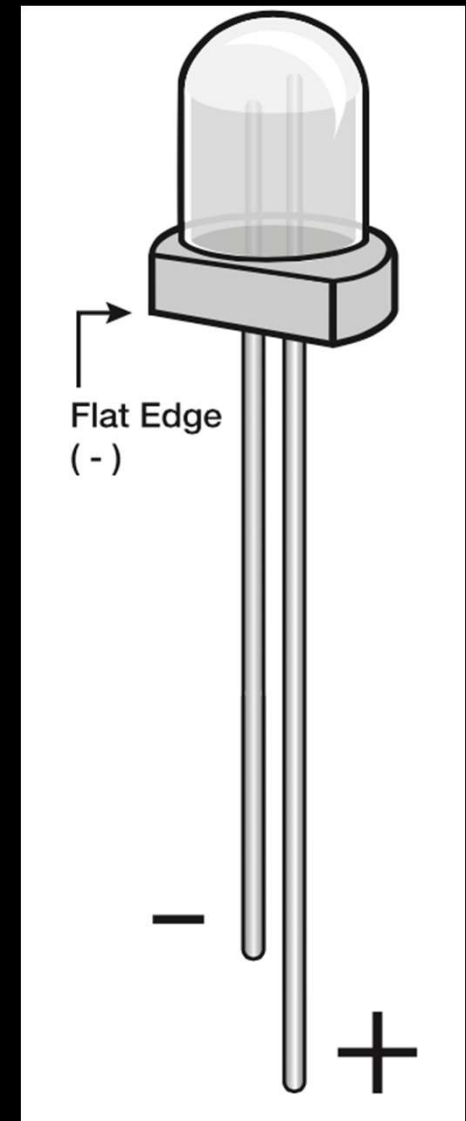
- Remove the LED from PIN 13 and GND
- Another LED on the board should start blinking
- The “**L**” on the Uno stands for LED
- Do you see this?



Blink an LED:

- Say you wanted to blink an LED on Pin 9, what would you change in the code?

- **int led = 9;**



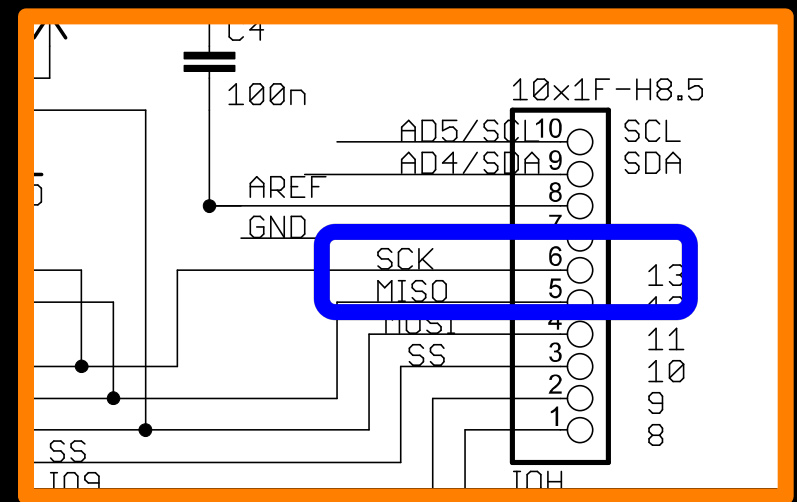
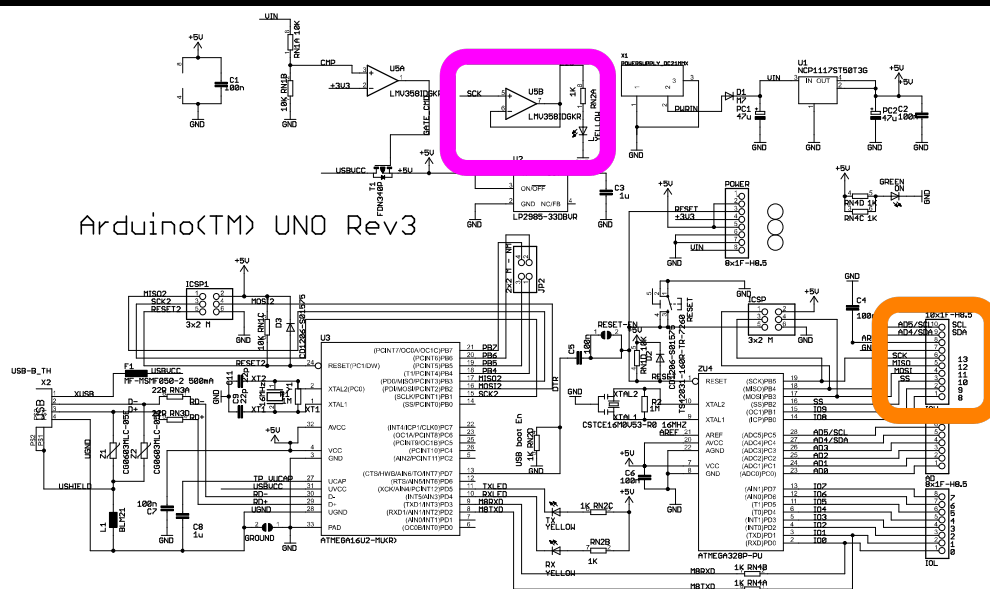
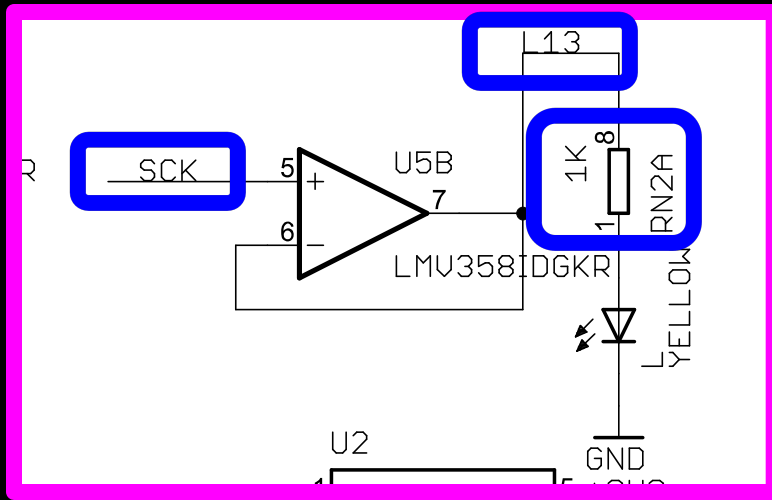
Blink an LED:

- **Could you connect LED directly to Pin 9 and GND like for Pin 13?**
- **No (OK for a few seconds) but why?**
- **LED requires some current limiting (resistor)**

Blink an LED:

- Let's look at Pin 13 on the schematic

- Follow the line and find a built in 1K resistor



Blink an LED:

- If you can Blink an LED, you can change the world
- Why?

**PLEASE SAVE YOUR
SKETCH FILE**





Arduino Driving Lesson

- A. Download Arduino IDE
- B. Arduino Overview
- C. Arduino Communication
- D. Blink an LED, Change the World

