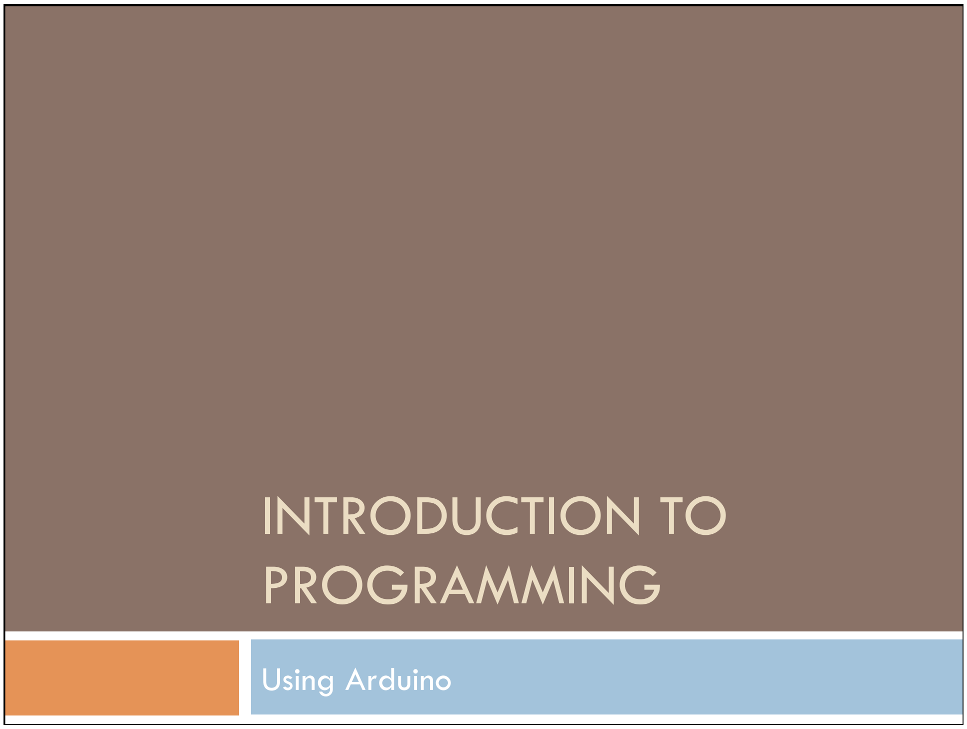
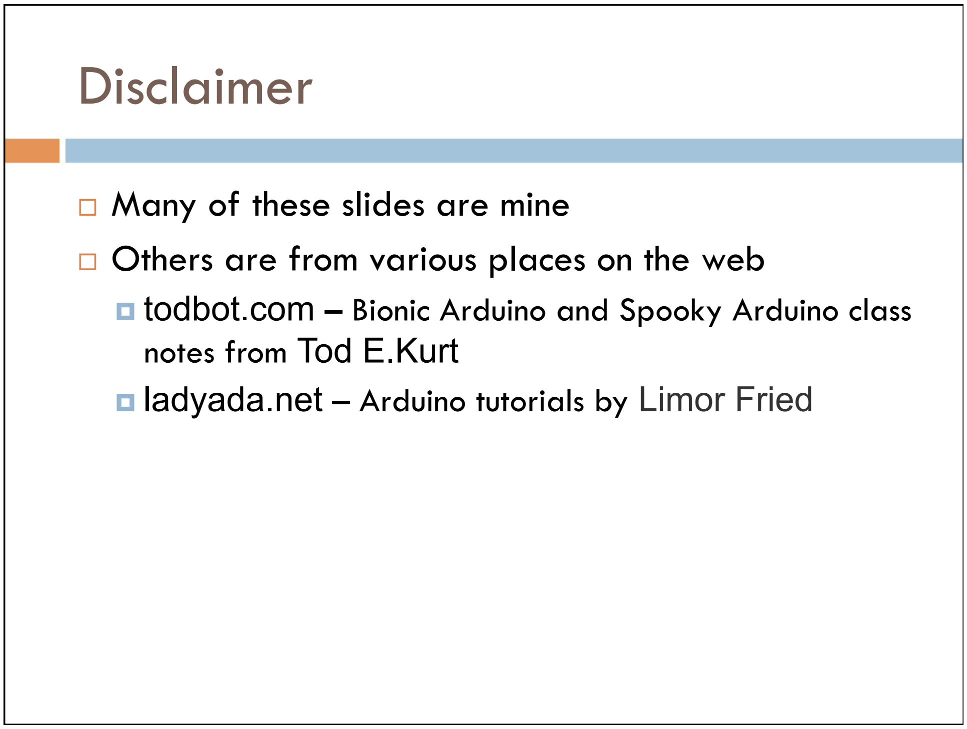


INTRODUCTION TO PROGRAMMING

Using Arduino



Disclaimer

- Many of these slides are mine
- Others are from various places on the web
  - todbot.com – Bionic Arduino and Spooky Arduino class notes from Tod E.Kurt
  - ladyada.net – Arduino tutorials by Limor Fried

What is a program?

- Essentially just a list of actions to take
  - Each line of the program is step to take
  - The program just walks through the steps one at at time
    - Maybe looping too
- It's like a recipe!

Meatloaf…

Meatloaf Recipe
Ingredients:

1 package Lipton Onion Soup Mix
2 pounds lean ground beef
1 large egg
2/3 cup milk
3 Tablespoons catsup
3 Tablespoons brown sugar
1 Tablespoon yellow mustard

Meatloaf…

Directions:

1. Preheat the oven to 350 degrees F.

2. Mix the onion soup mix, ground beef, egg and milk together.

3. Form the combination into a loaf shape in a 13 X 9 X 2 loaf pan.

4. Combine the rest of the ingredients and spoon onto the top of the meatloaf.

5. Bake uncovered, for about an hour.

6. When done, take the meatloaf out of the pan and place on a serving plate. Let stand for 10 minutes before slicing.

Shampoo

1. Lather
2. Rinse
3. Repeat

- □ When do you stop?

Shampoo

1. Lather
2. Rinse
3. If this is the first lather, then Repeat
   else stop and towel off

Shampoo

1. Repeat twice {
2. Lather
3. Rinse
4. }

Shampoo

```
For (count=1; count<3; count=count+1)
{
    Lather
    Rinse
}
```

Shampoo

```
For (count=1; count<3; count=count+1)
{
    Lather
    Rinse
}
```

```
count=1
lather
rinse
count=2
lather
rinse
count=3
continue to next instruction…
```

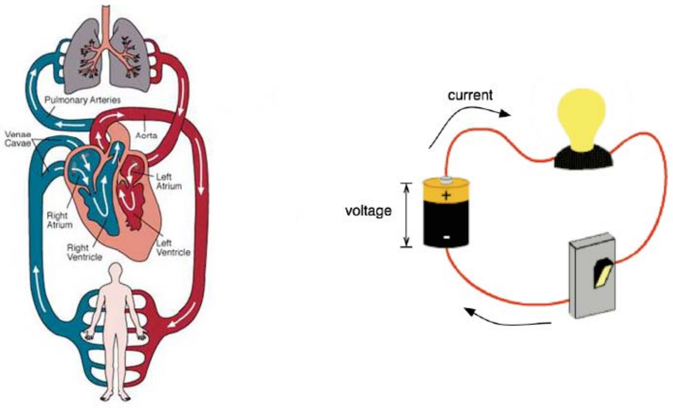
Make a light flash

1. Turn light on
2. Wait for 1 second
3. Turn light off
4. Wait for one second
5. repeat

□ We’ll come back to this… Let’s talk about lights

Electricity

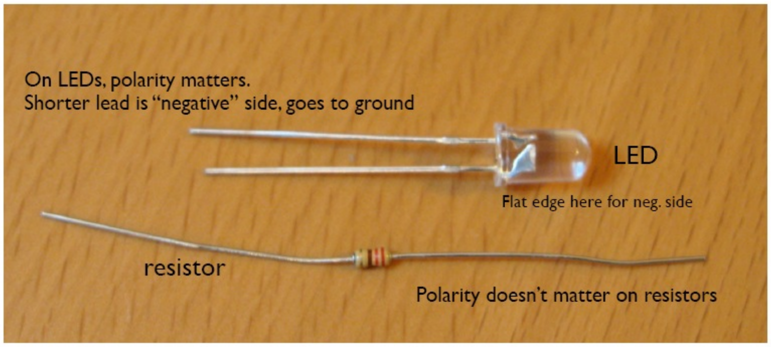
Making Circuits



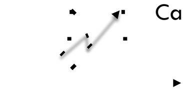
heart pumps, blood flows voltage pushes, current flows

www.todbot.com

LEDs and Resistors


On LEDs, polarity matters.
Shorter lead is "negative" side, goes to ground
LED
Flat edge here for neg. side
resistor
Polarity doesn't matter on resistors

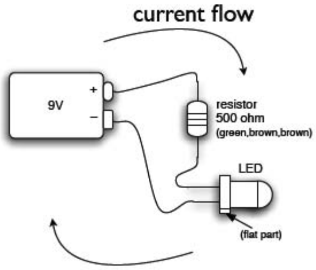
long lead
Anode +



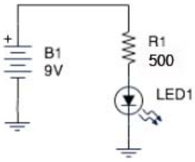
short lead
Cathode -

Current flows from Anode to Cathode
Lights up when current flows

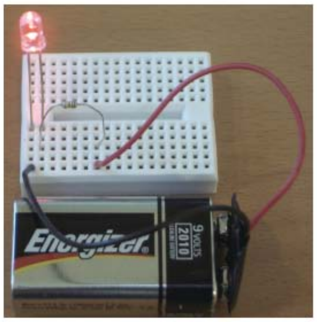
Wiring it up



wiring diagram



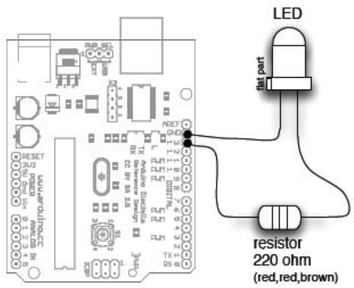
schematic



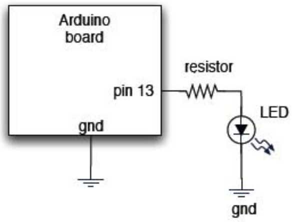
wiring it up

Electricity flows in a loop. Can stop flow by breaking the loop

Wiring it Up

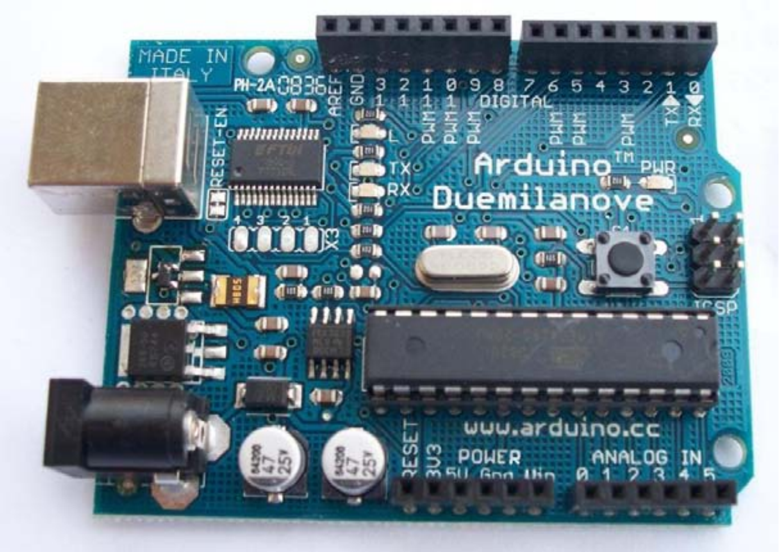


wiring diagram



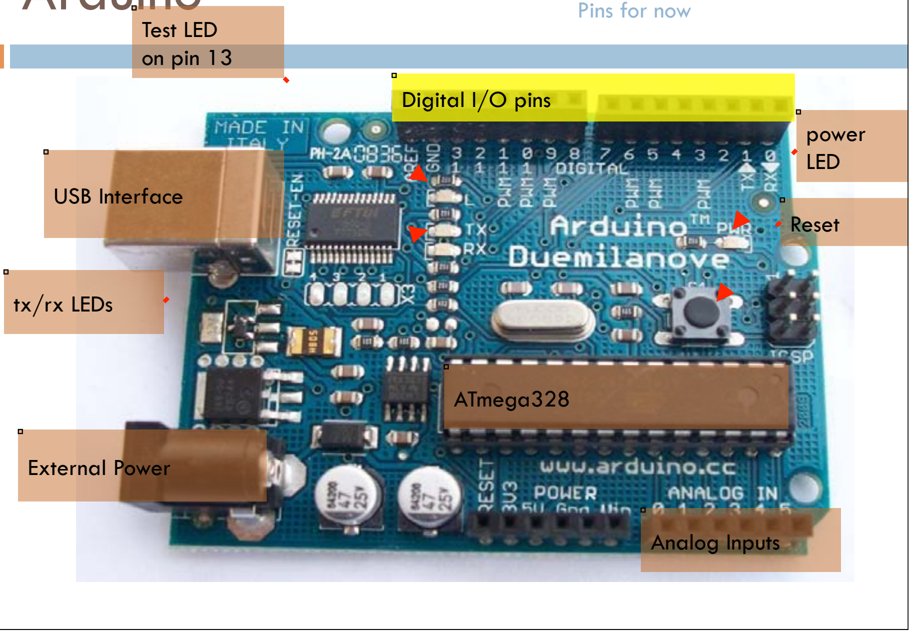
schematic

Arduino



Arduino

Focus on these Digital Pins for now



Test LED on pin 13

Digital I/O pins

power LED

USB Interface

Reset

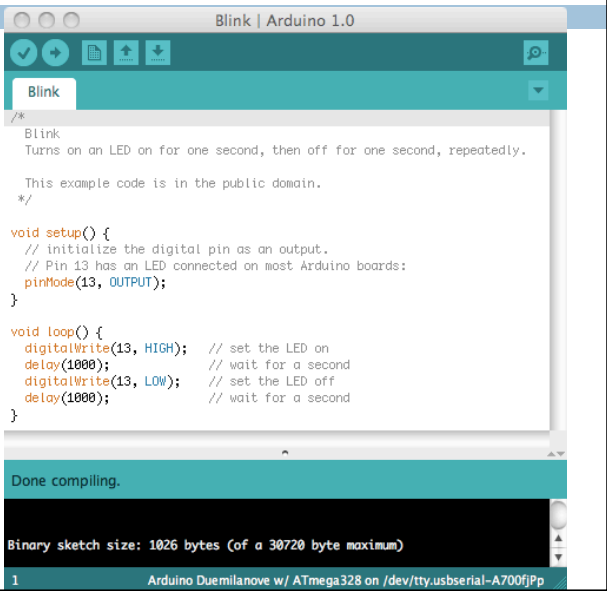
tx/rx LEDs

ATmega328

External Power

Analog Inputs

Arduino Programming



```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.
 */

void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH);   // set the LED on
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // set the LED off
  delay(1000);              // wait for a second
}
```

Done compiling.

```
Binary sketch size: 1026 bytes (of a 30720 byte maximum)
```

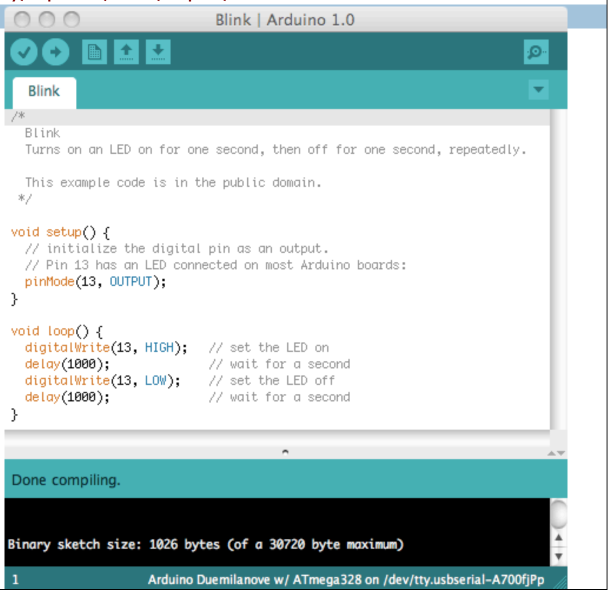
1 Arduino Duemilanove w/ ATmega328 on /dev/tty.usbserial-A700fjPp

Arduino Programming

Verify, Upload, New, Open, Save

Programming area

Notification area



```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.
 */

void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH);   // set the LED on
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // set the LED off
  delay(1000);              // wait for a second
}
```

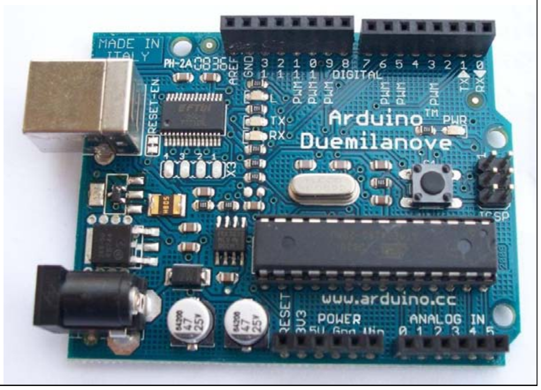
Done compiling.

```
Binary sketch size: 1026 bytes (of a 30720 byte maximum)
```

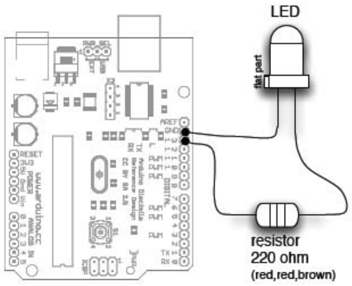
1 Arduino Duemilanove w/ ATmega328 on /dev/tty.usbserial-A700fjPp

Digital Pins

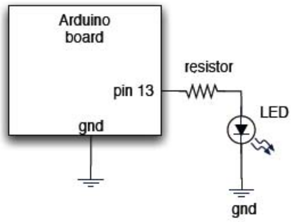
- Each of the digital pins can be set to one of two values
  - High and Low (+5v and 0v)
  - digitalWrite(<pin-number>, <value>);
  - digitalWrite(13, HIGH);
  - digitalWrite(13, LOW);



Wiring it Up



wiring diagram

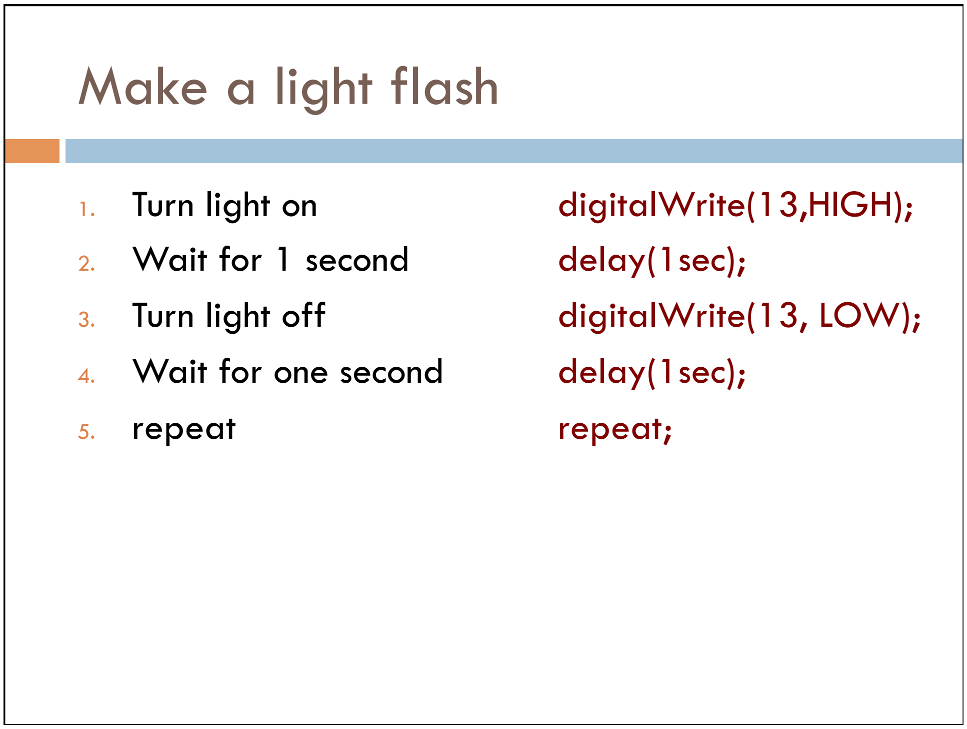


schematic

Arduino Duemilanove board has this circuit built-in
To turn on LED use `digitalWrite(13,HIGH)`

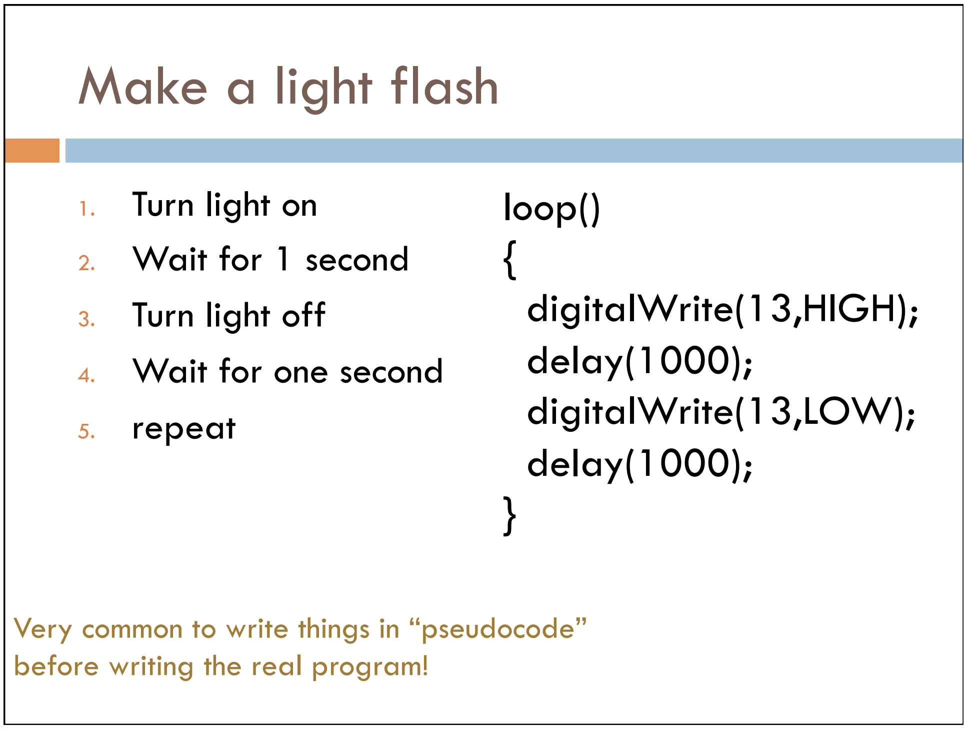
Make a light flash

1. Turn light on
2. Wait for 1 second
3. Turn light off
4. Wait for one second
5. repeat



Make a light flash

1.	Turn light on	digitalWrite(13,HIGH);
2.	Wait for 1 second	delay(1sec);
3.	Turn light off	digitalWrite(13, LOW);
4.	Wait for one second	delay(1sec);
5.	repeat	repeat;

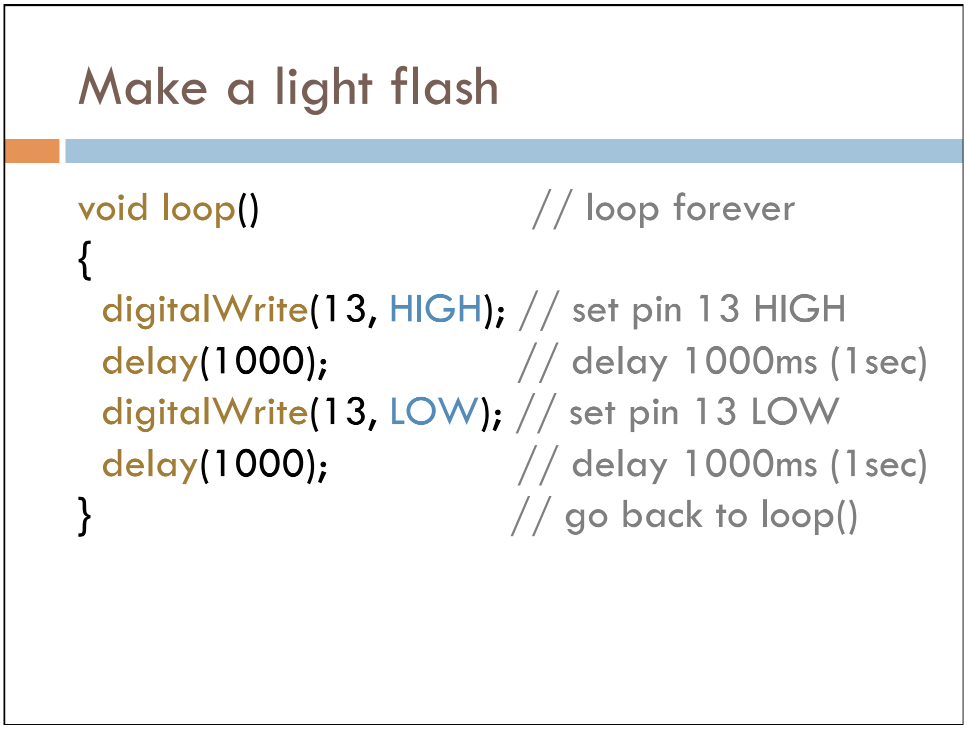


Make a light flash

1. Turn light on
2. Wait for 1 second
3. Turn light off
4. Wait for one second
5. repeat

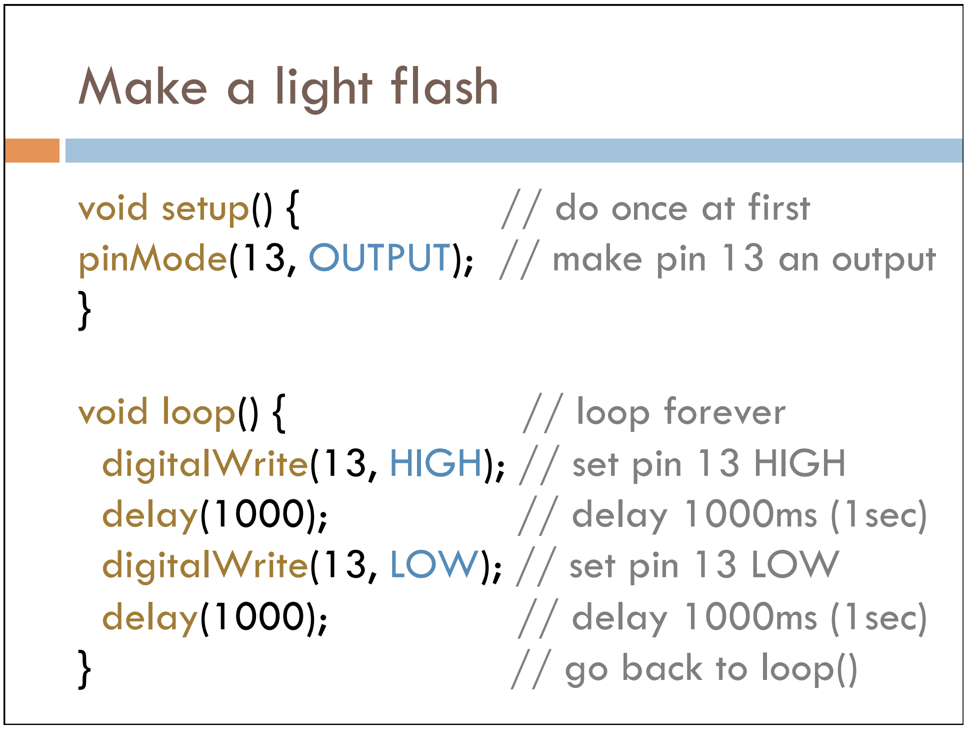
```
loop()
{
  digitalWrite(13,HIGH);
  delay(1000);
  digitalWrite(13,LOW);
  delay(1000);
}
```

Very common to write things in “pseudocode”
before writing the real program!



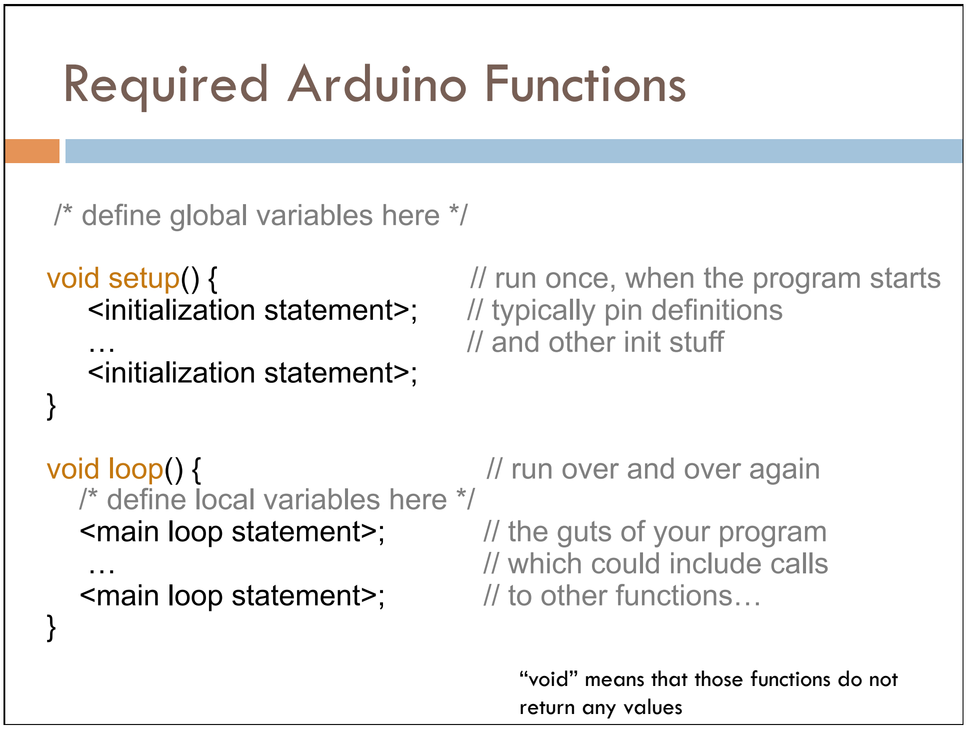
Make a light flash

```
void loop()                    // loop forever
{
  digitalWrite(13, HIGH); // set pin 13 HIGH
  delay(1000);            // delay 1000ms (1sec)
  digitalWrite(13, LOW);  // set pin 13 LOW
  delay(1000);            // delay 1000ms (1sec)
}                         // go back to loop()
```



Make a light flash

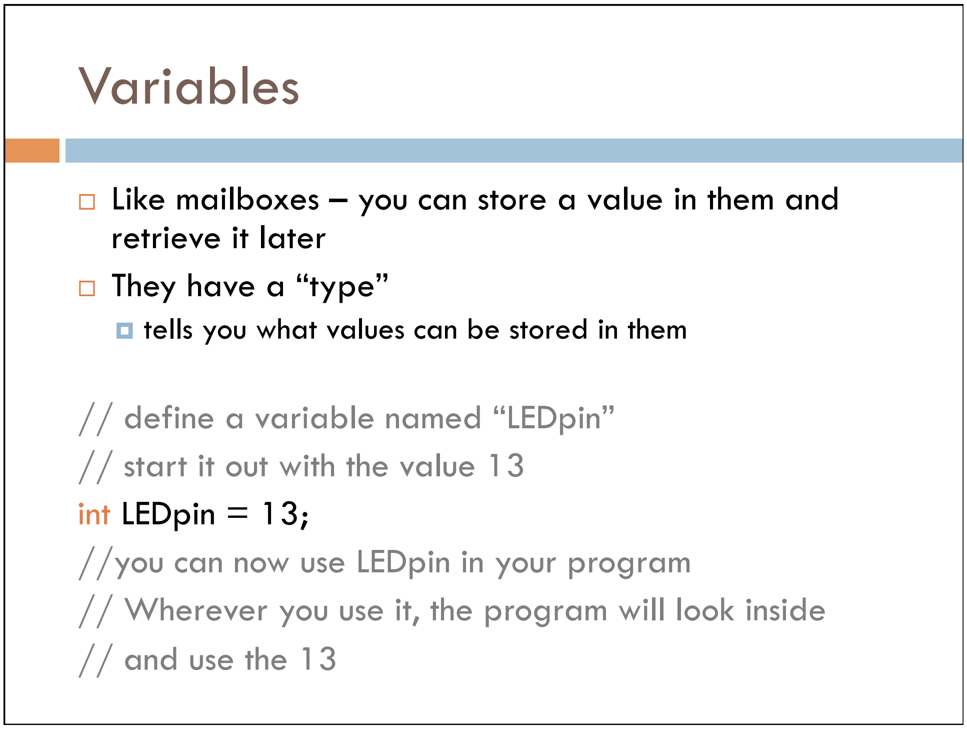
```
void setup() {              // do once at first
pinMode(13, OUTPUT);  // make pin 13 an output
}

void loop() {                  // loop forever
  digitalWrite(13, HIGH); // set pin 13 HIGH
  delay(1000);            // delay 1000ms (1sec)
  digitalWrite(13, LOW);  // set pin 13 LOW
  delay(1000);            // delay 1000ms (1sec)
}                         // go back to loop()
```



Required Arduino Functions

```
/* define global variables here */

void setup() {                    // run once, when the program starts
    <initialization statement>;   // typically pin definitions
    …                             // and other init stuff
    <initialization statement>;
}

void loop() {                     // run over and over again
   /* define local variables here */
   <main loop statement>;         // the guts of your program
    …                             // which could include calls
   <main loop statement>;         // to other functions…
}
```

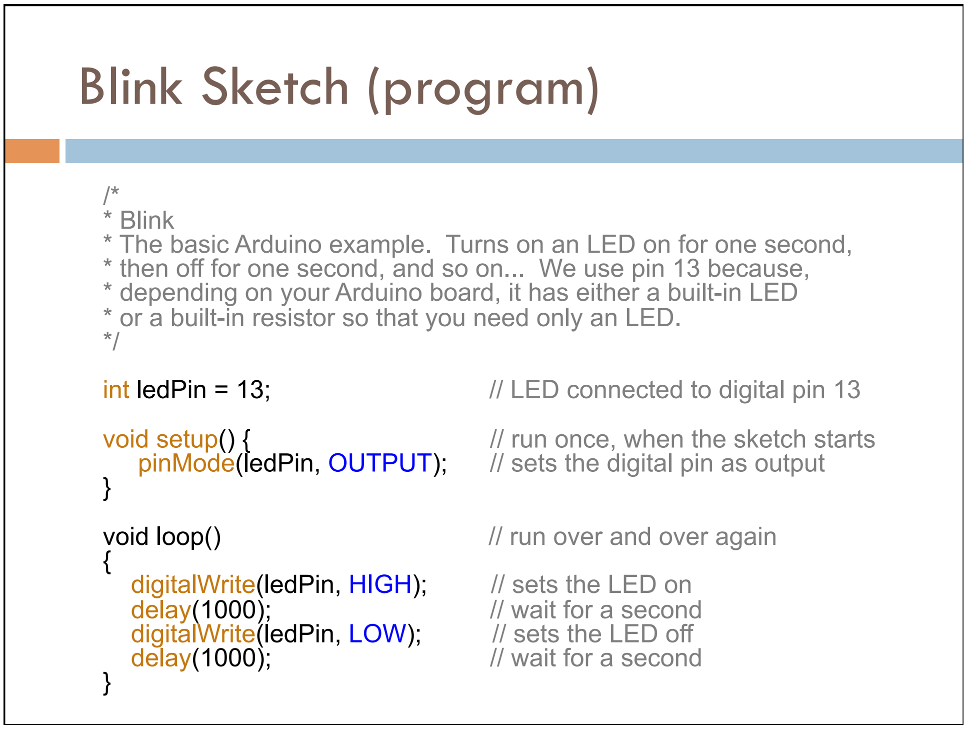
"void" means that those functions do not return any values



Variables

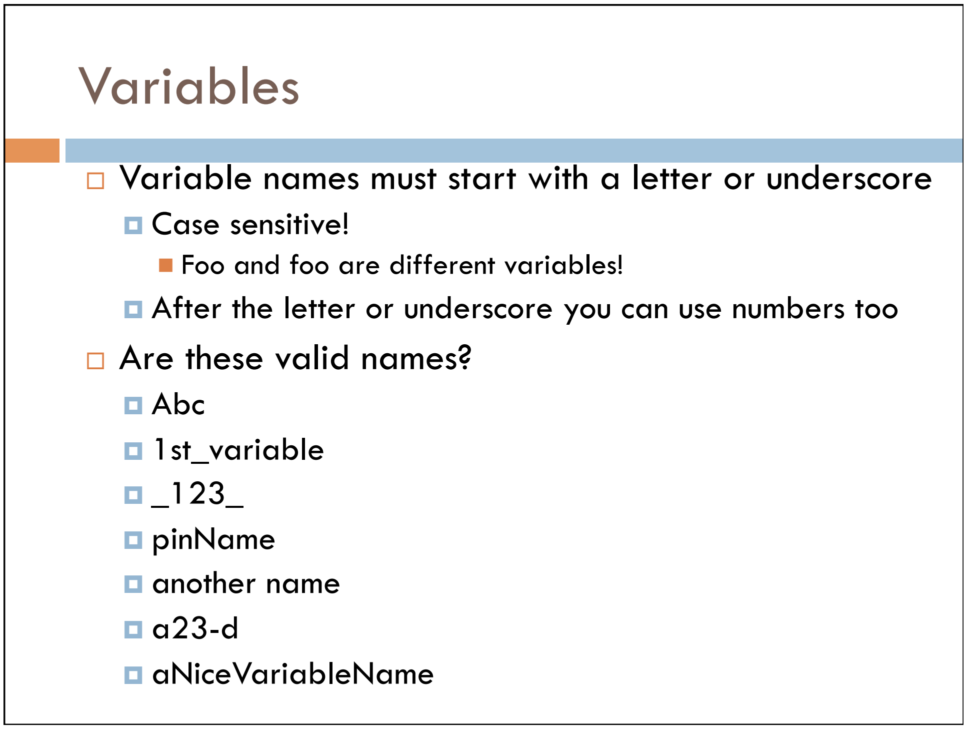
- Like mailboxes – you can store a value in them and retrieve it later
- They have a "type"
  - tells you what values can be stored in them

```
// define a variable named "LEDpin"
// start it out with the value 13
int LEDpin = 13;
//you can now use LEDpin in your program
// Wherever you use it, the program will look inside
// and use the 13
```



Blink Sketch (program)

```
/*
* Blink
* The basic Arduino example.  Turns on an LED on for one second,
* then off for one second, and so on...  We use pin 13 because,
* depending on your Arduino board, it has either a built-in LED
* or a built-in resistor so that you need only an LED.
*/

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

void setup() {                      // run once, when the sketch starts
    pinMode(ledPin, OUTPUT);        // sets the digital pin as output
}

void loop()                         // run over and over again
{
   digitalWrite(ledPin, HIGH);      // sets the LED on
   delay(1000);                     // wait for a second
   digitalWrite(ledPin, LOW);       // sets the LED off
   delay(1000);                     // wait for a second
}
```

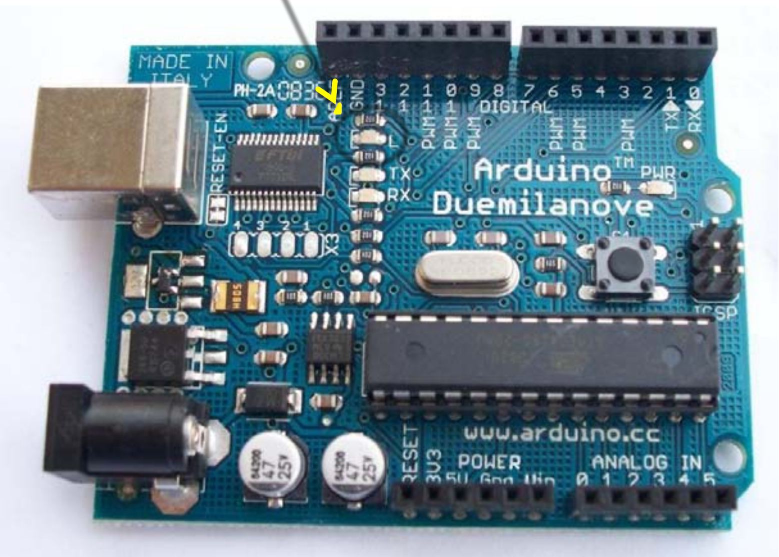


Variables

- Variable names must start with a letter or underscore
  - Case sensitive!
    - Foo and foo are different variables!
  - After the letter or underscore you can use numbers too
- Are these valid names?
  - Abc
  - 1st_variable
  - _123_
  - pinName
  - another name
  - a23-d
  - aNiceVariableName

Arduino

LED



Blink Modifications

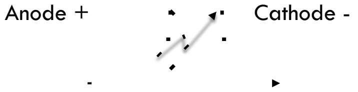
- □ Change so that blink is on for 500msec and off for 100msec
  - ◘ What happens?
- □ Change so that blink is on for 50msec and off for 50msec
  - ◘ What happens?
- □ Change so that blink is on for 10ms and off for 10ms
  - ◘ What happens?

We just made an LED blink…Big Deal?

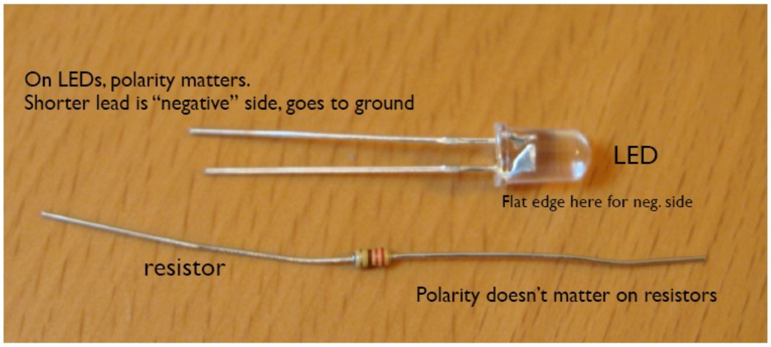
- Most actuators are switched on and off with a digital output
  - The digitalWrite(pin,value); function is the software command that lets you control almost anything
- LEDs are easy!
  - Motors, servos, etc. are a little trickier, but not much
  - More on that later…
- Arduino has 14 digital pins (inputs or outputs)
  - can easily add more with external helper chips
  - More on that later…

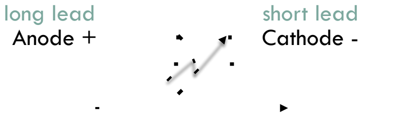
Blink Modifications

- Change to use an external LED rather than the one on the board
  - Connect to any digital pin
  - LED is on if current flows from Anode to Cathode
  - LED is on if the digital pin is HIGH, off if LOW
  - How much current do you use?
    - not more than 20mA
  - How do you make sure you don't use too much?
    - use a resistor
  - Pay attention to current! Use a current-limiting resistor!



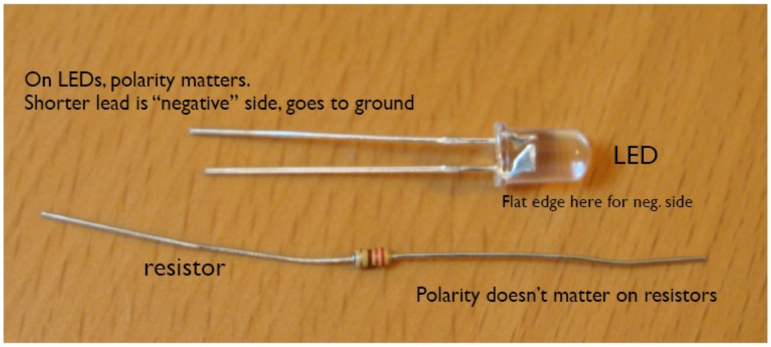
LEDs and Resistors

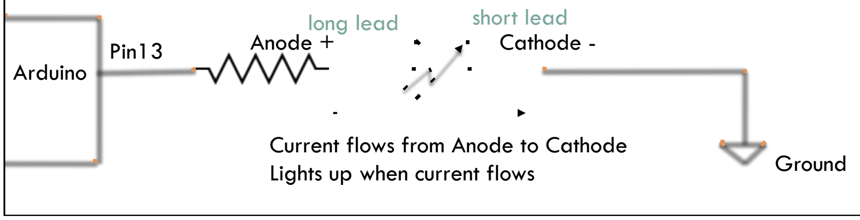

On LEDs, polarity matters.
Shorter lead is "negative" side, goes to ground
LED
Flat edge here for neg. side
resistor
Polarity doesn't matter on resistors



Current flows from Anode to Cathode
Lights up when current flows

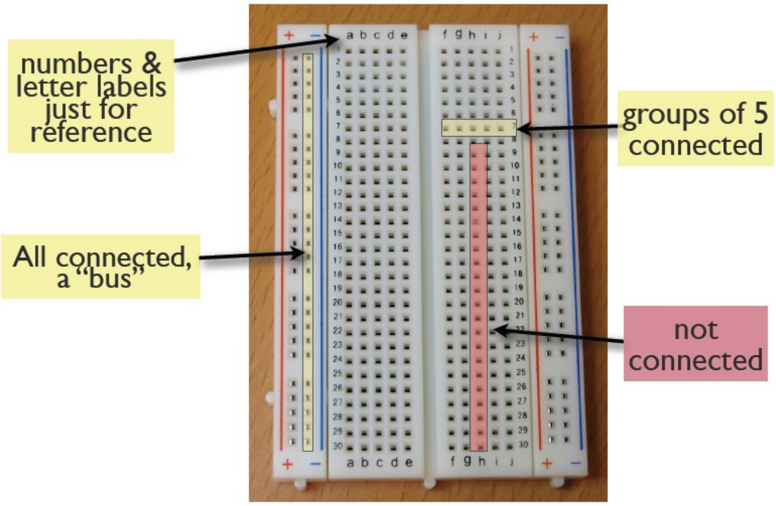
LEDs and Resistors


On LEDs, polarity matters.
Shorter lead is "negative" side, goes to ground
LED
Flat edge here for neg. side
resistor
Polarity doesn't matter on resistors



Current flows from Anode to Cathode
Lights up when current flows

Proto Boards



numbers & letter labels just for reference

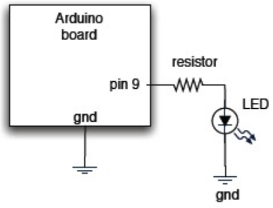
All connected, a "bus"

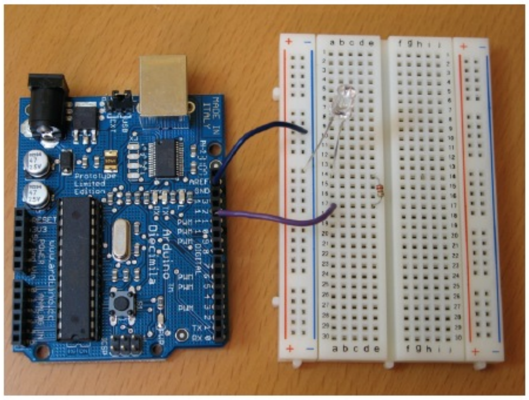
groups of 5 connected

not connected

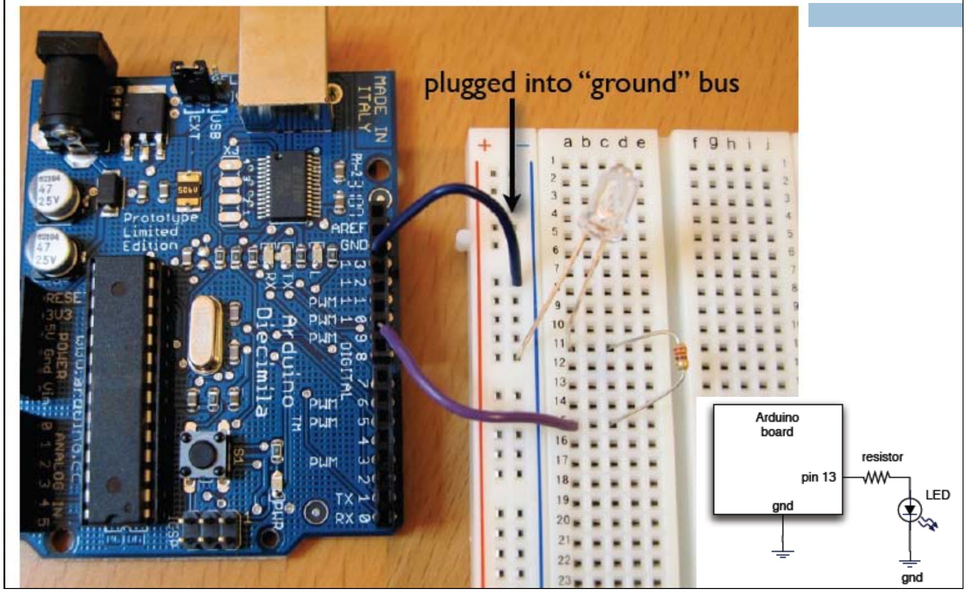
AKA Solderless Breadboards

Wire it Up



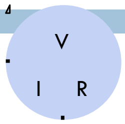


Wire it Up



Current Limiting Resistor

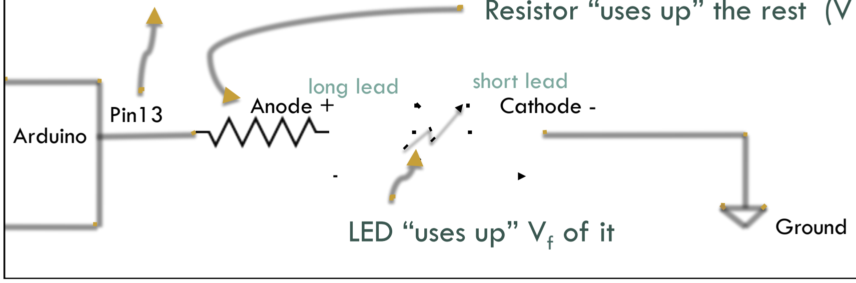
- Ohm's Law
  - V = IR   I = V/R R = V/I



- Every LED has a V_f "Forward Voltage"
  - How much voltage is dropped (used up) passing through the LED

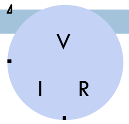
"HIGH" forces output pin to 5v (called V)

Resistor "uses up" the rest $(V - V_f)$



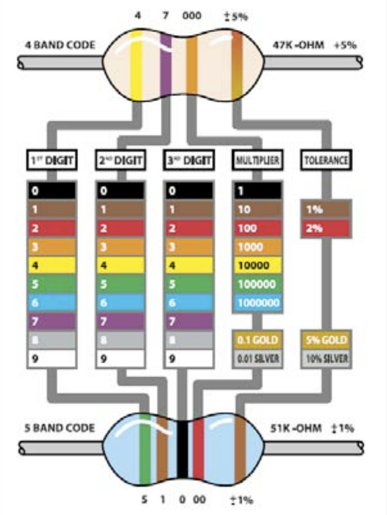
LED "uses up" V_f of it

Current Limiting Resistor



- Ohm's Law
  - $V = IR \quad I = V/R \quad R = V/I$
- Every LED has a V_f "Forward Voltage"
  - How much voltage is dropped (used up) passing through the LED
- $R = (V - V_f) / I$
  - Example – If V_f is 1.9v (red LED), and V = 5v, and you want 15mA of current (0.015A)
  - $R = (5 - 1.9)/0.015 = 3.1/0.015 = 206\Omega$
  - Exact isn't critical – use next size up, i.e. 220Ω
  - Or be safe and use 330Ω or 470Ω
  - This would result in 9.4mA or 6.6mA which is fine

Resistor Color Codes



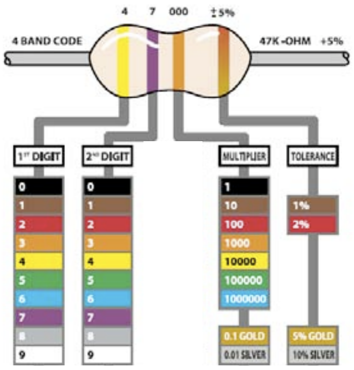
1ST DIGIT	2ND DIGIT	3RD DIGIT	MULTIPLIER	TOLERANCE
0	0	0	1	
1	1	1	10	1%
2	2	2	100	2%
3	3	3	1000	
4	4	4	10000	
5	5	5	100000	
6	6	6	1000000	
7	7	7		
8	8	8	0.1 GOLD	5% GOLD
9	9	9	0.01 SILVER	10% SILVER

What's the color code for a 330Ω resistor?

What's the color code for a 1kΩ resistor?

What's the color code for a 10kΩ resistor

Resistor Color Codes



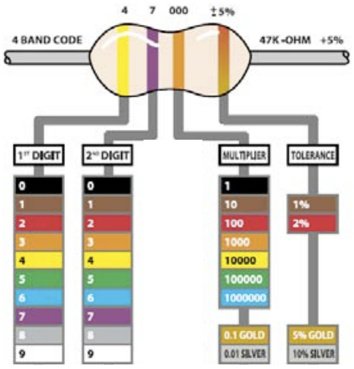
We're using 4-band 5% resistors with a ¼ watt rating

What's the color code for a 330Ω resistor?

What's the color code for a 1kΩ resistor?

What's the color code for a 10kΩ resistor

Resistor Color Codes



We're using 4-band 5% resistors with a ¼ watt rating

What's the color code for a 330Ω resistor?

orange orange brown gold

What's the color code for a 1kΩ resistor?

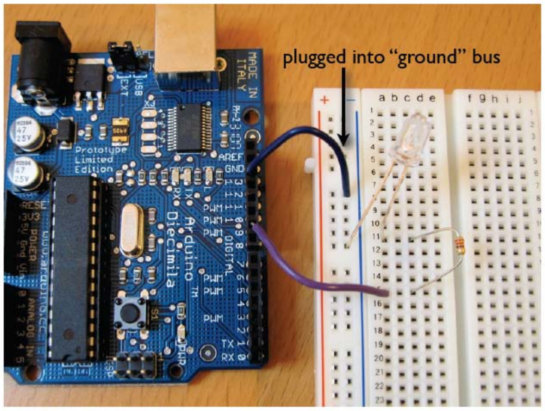
brown black red gold

What's the color code for a 470Ω resistor

brown black orange gold

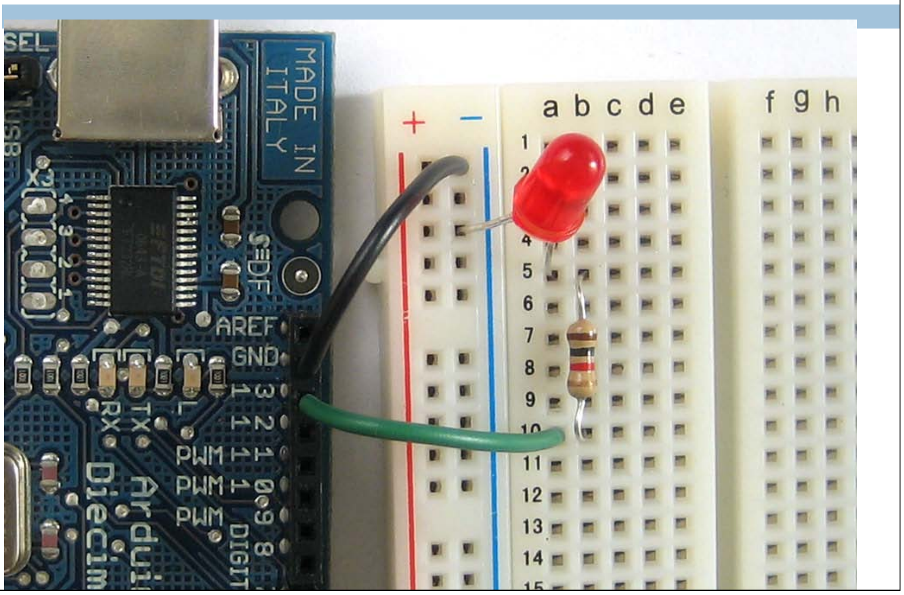
Wire it Up

- Wire up an external LED of your choice, and change the Blink program to use that external LED
  - Choose your resistor based on the V_f of the LED you're using
    - Usually 1.8-2.2v
    - Listed on class web site
  - If you don't know Vf pick 330Ω or 470Ω



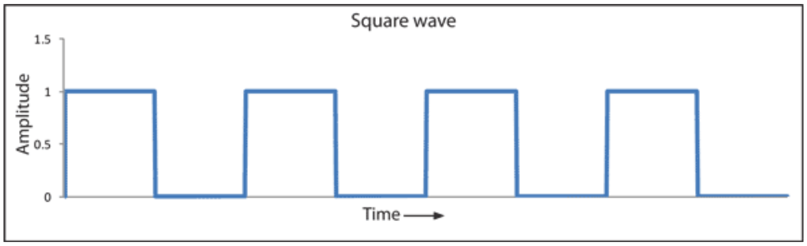
plugged into “ground” bus

Another view



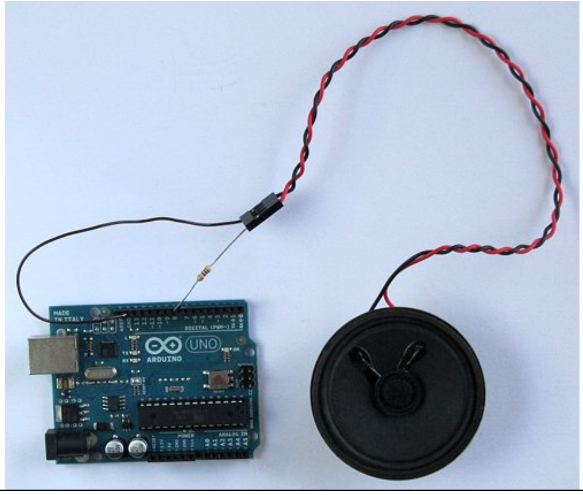
Sound!

- Now – how do we make noise with an Arduino?
  - Use a digital output
  - Flip it up and down at an audio frequency



Sound!

- Now connect a speaker to that pin…



Why include a resistor?

- Ohm's Law: V = IR   or R = V/I
- Arduino pins can provide, or consume 40mA
  - 40mA is 0.040A   (1 mA is 1/1000 of an Amp)
- So, 5v/0.040A = 125Ω
  - Or, if you want to be safe, 5v/0.035A = 143Ω
- Speakers are typically 8Ω
  - 125-8 = 117Ω   143-8= 135Ω

This is a "current limiting resistor"

Make sound from a program

- The Arduino function that makes a sound is
  - tone(<pin>, <freq-in-Hertz>);
  - tone(<pin>, <freq-in-Hertz>, <duration-in-ms>);
- Examples:

```
tone(10, 440);  // play a 440Hz tone on pin 10
noTone(10);     // stop playing the tone on pin 10

int myPin = 9;        // define a variable named myPin
int myTone = 440; // another named myTone
tone(myPin, myTone, 1000); // play for 1sec
```

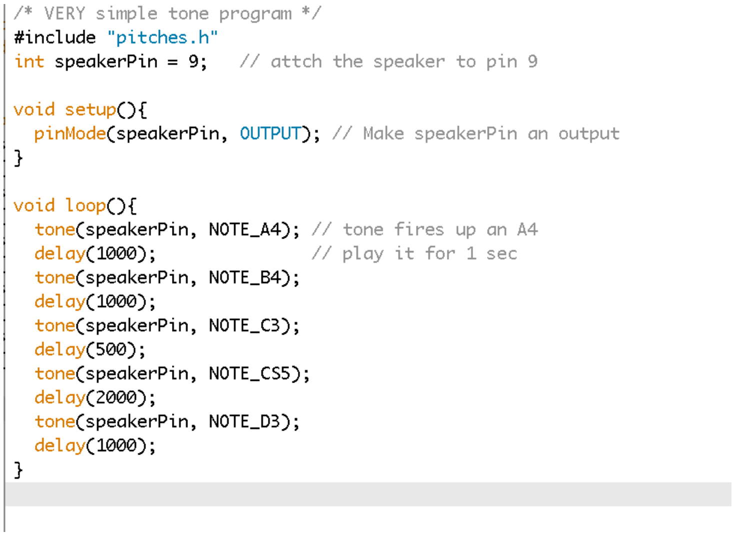
Standard pitches – “pitches.h”

- □ Codifies “standard” pitches

```
#define NOTE_FS4 370
#define NOTE_G4  392
#define NOTE_GS4 415
#define NOTE_A4  440
#define NOTE_AS4 466
#define NOTE_B4  494
```

```
tone(myPin, NOTE_A4); // play standard A above middle C
```

Example

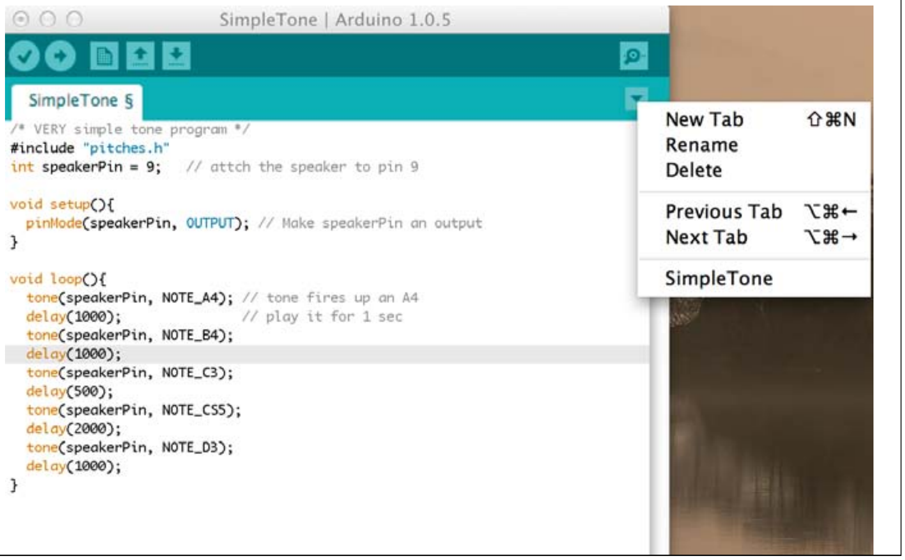


```
/* VERY simple tone program */
#include "pitches.h"
int speakerPin = 9;   // attch the speaker to pin 9

void setup(){
  pinMode(speakerPin, OUTPUT); // Make speakerPin an output
}

void loop(){
  tone(speakerPin, NOTE_A4); // tone fires up an A4
  delay(1000);               // play it for 1 sec
  tone(speakerPin, NOTE_B4);
  delay(1000);
  tone(speakerPin, NOTE_C3);
  delay(500);
  tone(speakerPin, NOTE_CS5);
  delay(2000);
  tone(speakerPin, NOTE_D3);
  delay(1000);
}
```

Getting the pitches.h file

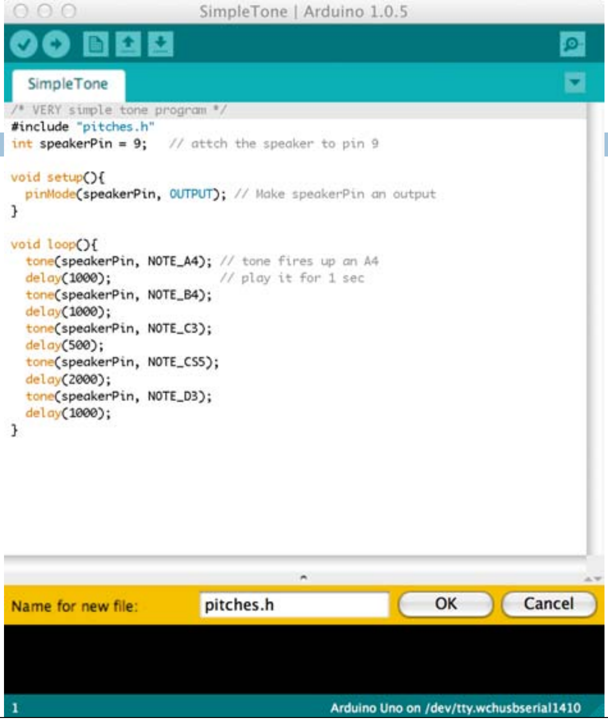
- □ Go to http://arduino.cc/en/Tutorial/Tone
- □ Copy the pitch data
- □ In the Arduino editor make a new “tab”
- □ Name the tab “pitches.h”
- □ Paste in the data

Make a new tab



```
/* VERY simple tone program */
#include "pitches.h"
int speakerPin = 9;   // attch the speaker to pin 9

void setup(){
  pinMode(speakerPin, OUTPUT); // Make speakerPin an output
}

void loop(){
  tone(speakerPin, NOTE_A4); // tone fires up an A4
  delay(1000);               // play it for 1 sec
  tone(speakerPin, NOTE_B4);
  delay(1000);
  tone(speakerPin, NOTE_C3);
  delay(500);
  tone(speakerPin, NOTE_CS5);
  delay(2000);
  tone(speakerPin, NOTE_D3);
  delay(1000);
}
```

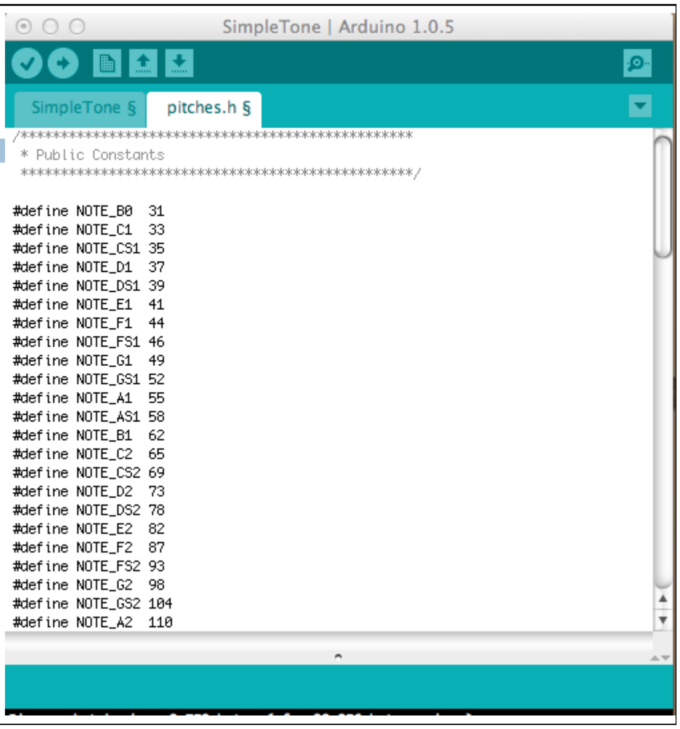
Name it pitches.h


SimpleTone | Arduino 1.0.5
SimpleTone
OK
Cancel
1
Arduino Uno on /dev/tty.wchusbserial1410

```
/* VERY simple tone program */
#include "pitches.h"
int speakerPin = 9;   // attch the speaker to pin 9

void setup(){
  pinMode(speakerPin, OUTPUT); // Make speakerPin an output
}

void loop(){
  tone(speakerPin, NOTE_A4); // tone fires up an A4
  delay(1000);               // play it for 1 sec
  tone(speakerPin, NOTE_B4);
  delay(1000);
  tone(speakerPin, NOTE_C3);
  delay(500);
  tone(speakerPin, NOTE_CS5);
  delay(2000);
  tone(speakerPin, NOTE_D3);
  delay(1000);
}
```

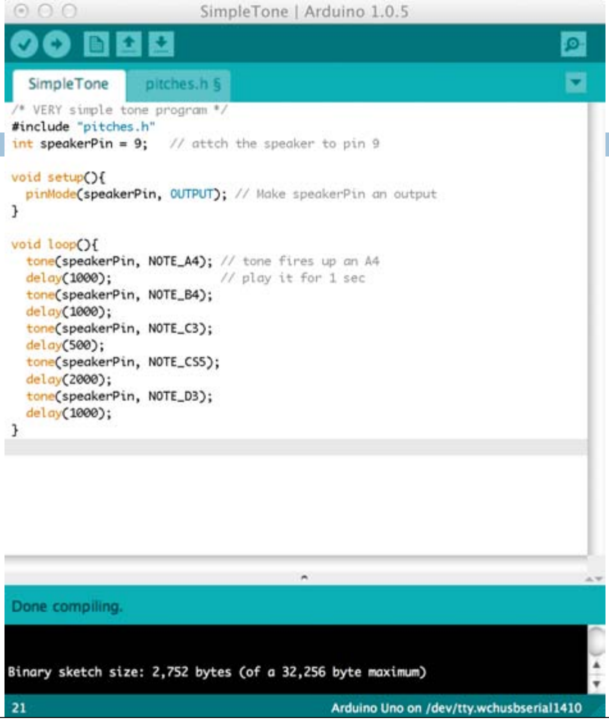
Name for new file: pitches.h

Add the data from the web site



```
/*************************************************
 * Public Constants
 *************************************************/

#define NOTE_B0  31
#define NOTE_C1  33
#define NOTE_CS1 35
#define NOTE_D1  37
#define NOTE_DS1 39
#define NOTE_E1  41
#define NOTE_F1  44
#define NOTE_FS1 46
#define NOTE_G1  49
#define NOTE_GS1 52
#define NOTE_A1  55
#define NOTE_AS1 58
#define NOTE_B1  62
#define NOTE_C2  65
#define NOTE_CS2 69
#define NOTE_D2  73
#define NOTE_DS2 78
#define NOTE_E2  82
#define NOTE_F2  87
#define NOTE_FS2 93
#define NOTE_G2  98
#define NOTE_GS2 104
#define NOTE_A2  110
```

Compile and run

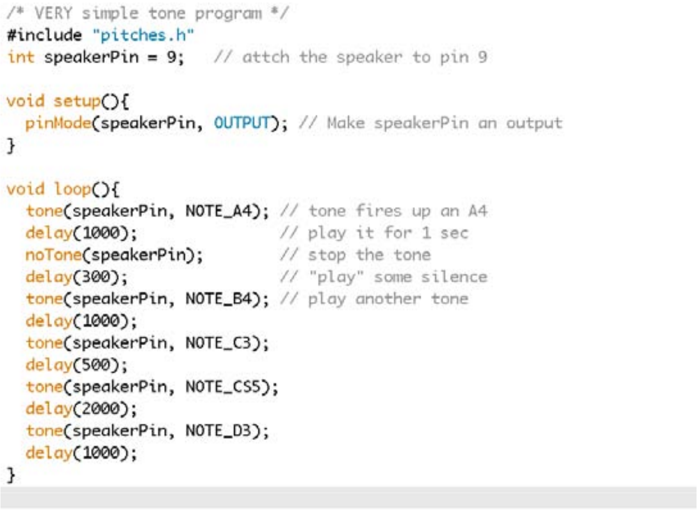


```
/* VERY simple tone program */
#include "pitches.h"
int speakerPin = 9;    // attch the speaker to pin 9

void setup(){
  pinMode(speakerPin, OUTPUT); // Make speakerPin an output
}

void loop(){
  tone(speakerPin, NOTE_A4); // tone fires up an A4
  delay(1000);               // play it for 1 sec
  tone(speakerPin, NOTE_B4);
  delay(1000);
  tone(speakerPin, NOTE_C3);
  delay(500);
  tone(speakerPin, NOTE_CS5);
  delay(2000);
  tone(speakerPin, NOTE_D3);
  delay(1000);
}
```

Done compiling.

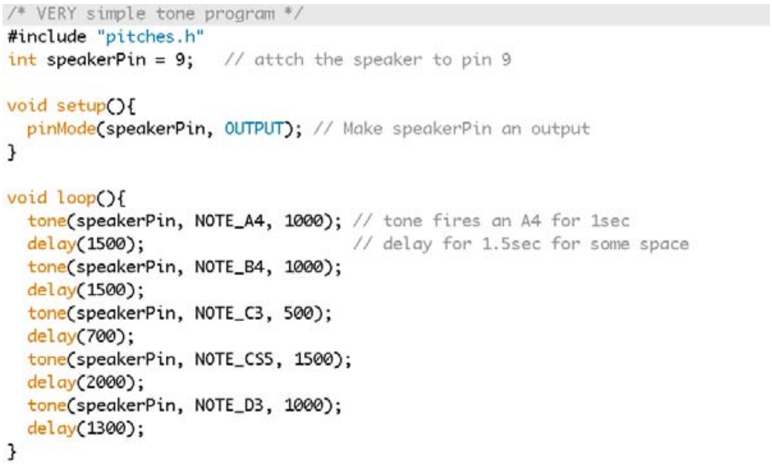
```
Binary sketch size: 2,752 bytes (of a 32,256 byte maximum)
```

Add space between notes



```
/* VERY simple tone program */
#include "pitches.h"
int speakerPin = 9;    // attch the speaker to pin 9

void setup(){
  pinMode(speakerPin, OUTPUT); // Make speakerPin an output
}

void loop(){
  tone(speakerPin, NOTE_A4); // tone fires up an A4
  delay(1000);               // play it for 1 sec
  noTone(speakerPin);        // stop the tone
  delay(300);                // "play" some silence
  tone(speakerPin, NOTE_B4); // play another tone
  delay(1000);
  tone(speakerPin, NOTE_C3);
  delay(500);
  tone(speakerPin, NOTE_CS5);
  delay(2000);
  tone(speakerPin, NOTE_D3);
  delay(1000);
}
```

Add space between notes



```
/* VERY simple tone program */
#include "pitches.h"
int speakerPin = 9;   // attch the speaker to pin 9

void setup(){
  pinMode(speakerPin, OUTPUT); // Make speakerPin an output
}

void loop(){
  tone(speakerPin, NOTE_A4, 1000); // tone fires an A4 for 1sec
  delay(1500);                     // delay for 1.5sec for some space
  tone(speakerPin, NOTE_B4, 1000);
  delay(1500);
  tone(speakerPin, NOTE_C3, 500);
  delay(700);
  tone(speakerPin, NOTE_CS5, 1500);
  delay(2000);
  tone(speakerPin, NOTE_D3, 1000);
  delay(1300);
}
```

Additional Programming

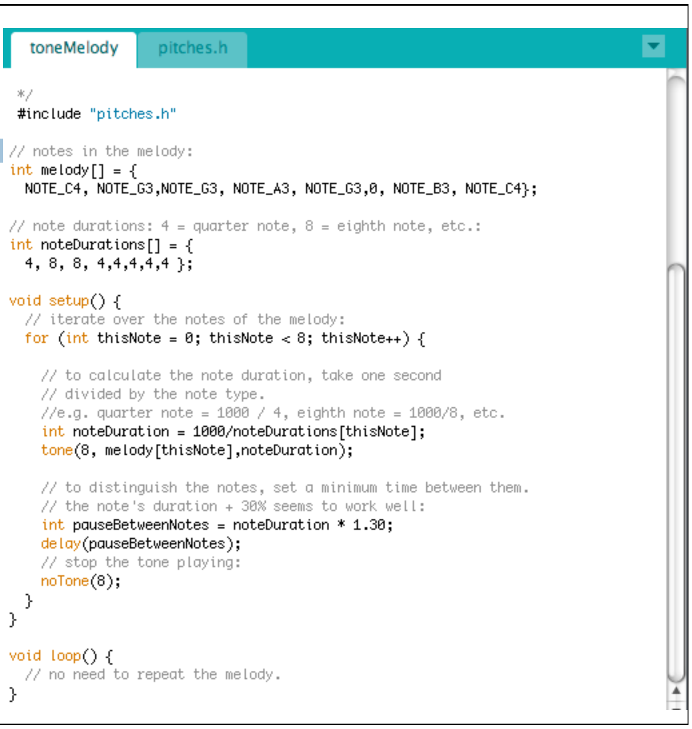
- Generate random number
  random(<min>,<max>)
  - Returns random number between min and max-1

random(2, 5); // returns random number between 2 and 4

Random

```
int myNum;     // variable to hold a number
myNum = random(1000, 2001); // between 1000, 2000

tone(9, myNum, 1000);  // play a random tone

tone(9, random(500, 1500));  // play another random tone

tone(9, 440, random(1000, 2000)); // play for a random duration
```

Example from Arduino

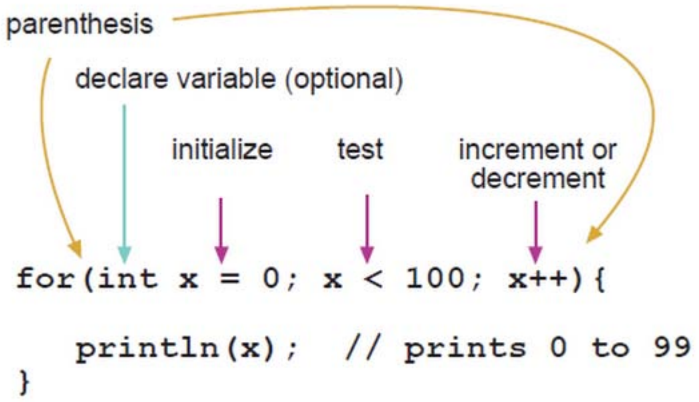


```
 */
 #include "pitches.h"

// notes in the melody:
int melody[] = {
  NOTE_C4, NOTE_G3,NOTE_G3, NOTE_A3, NOTE_G3,0, NOTE_B3, NOTE_C4};

// note durations: 4 = quarter note, 8 = eighth note, etc.:
int noteDurations[] = {
  4, 8, 8, 4,4,4,4,4 };

void setup() {
  // iterate over the notes of the melody:
  for (int thisNote = 0; thisNote < 8; thisNote++) {

    // to calculate the note duration, take one second
    // divided by the note type.
    //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
    int noteDuration = 1000/noteDurations[thisNote];
    tone(8, melody[thisNote],noteDuration);

    // to distinguish the notes, set a minimum time between them.
    // the note's duration + 30% seems to work well:
    int pauseBetweenNotes = noteDuration * 1.30;
    delay(pauseBetweenNotes);
    // stop the tone playing:
    noTone(8);
  }
}

void loop() {
  // no need to repeat the melody.
}
```

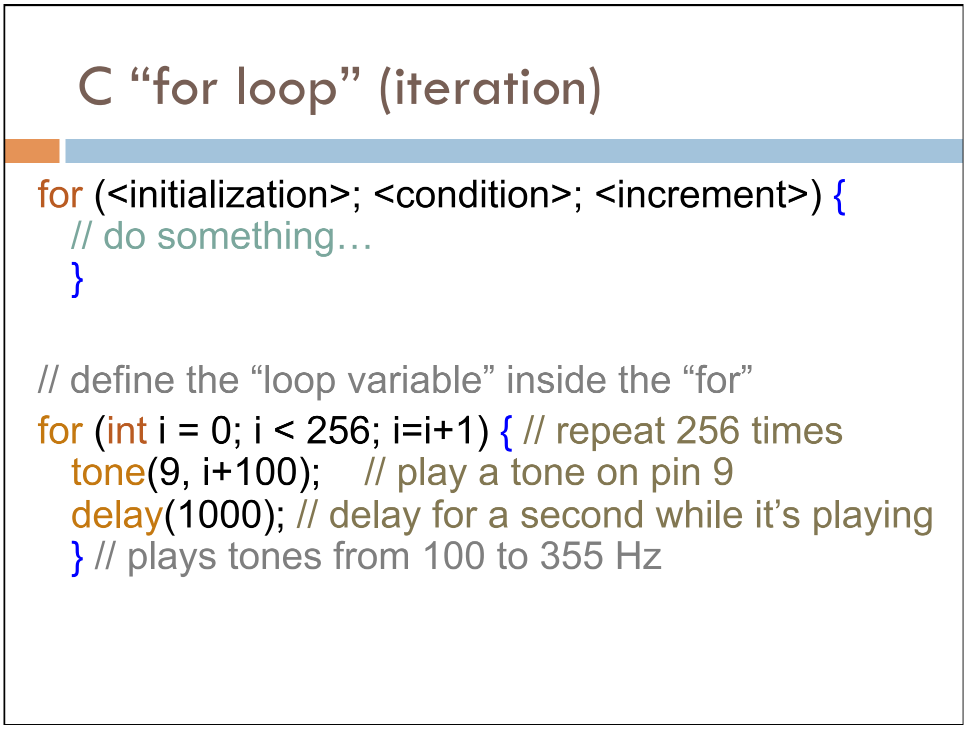
C “for loop” (iteration)

```
for (<initialization>; <condition>; <increment>) {
   // do something…
   }

int i;   // define an int to use as a loop variable
for (i = 0; i < 256; i=i+1) { // repeat 256 times
   tone(9, i+100);     // play a tone on pin 9
   delay(1000); // delay for a second while it’s playing
} // plays tones from 100 to 355 Hz
```

Another view of a “for loop”

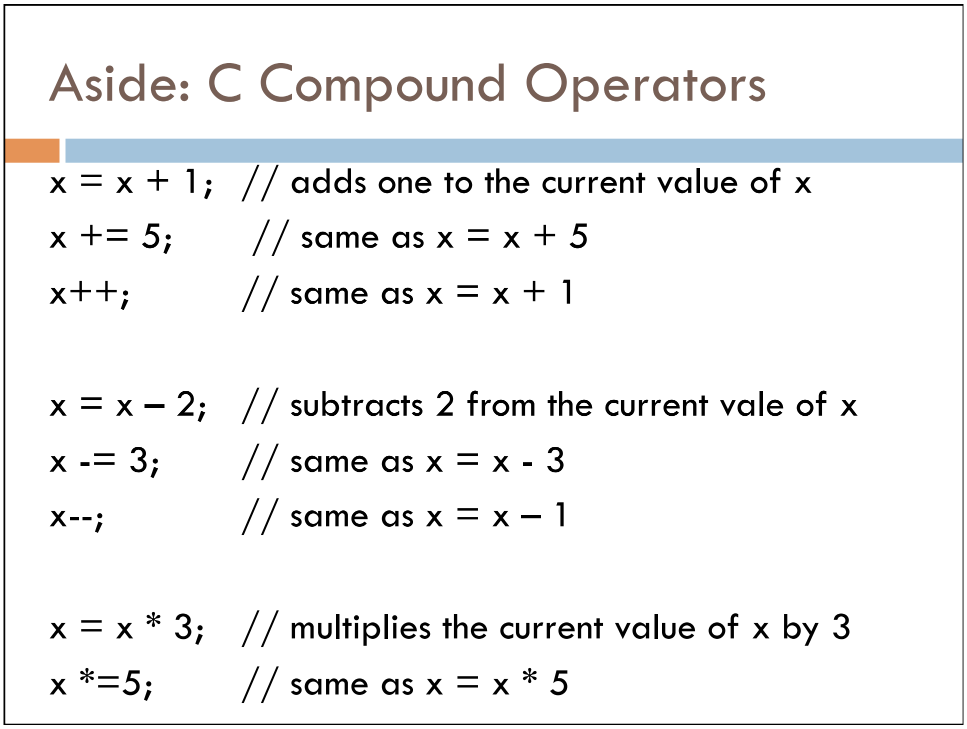


```
for(int x = 0; x < 100; x++){
    println(x);   // prints 0 to 99
}
```



C “for loop” (iteration)

```
for (<initialization>; <condition>; <increment>) {
   // do something…
   }

// define the “loop variable” inside the “for”
for (int i = 0; i < 256; i=i+1) { // repeat 256 times
   tone(9, i+100);    // play a tone on pin 9
   delay(1000); // delay for a second while it’s playing
   } // plays tones from 100 to 355 Hz
```



Aside: C Compound Operators

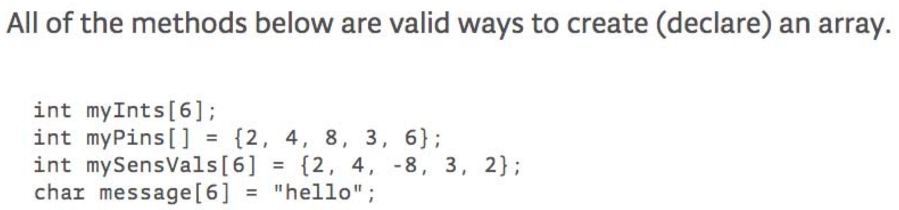
```
x = x + 1;  // adds one to the current value of x
x += 5;      // same as x = x + 5
x++;        // same as x = x + 1

x = x – 2;  // subtracts 2 from the current vale of x
x -= 3;     // same as x = x - 3
x--;        // same as x = x – 1

x = x * 3;  // multiplies the current value of x by 3
x *=5;      // same as x = x * 5
```

C “for loop” (iteration)

```
for (<initialization>; <condition>; <increment>) {
   // do something…
   }

// define the “loop variable” inside the “for”
for (int i = 0; i < 256; i++) { // repeat 256 times
   tone(9, i+100);    // play a tone on pin 9
   delay(1000); // delay for a second while it’s playing
} // plays tones from 100 to 355 Hz
```

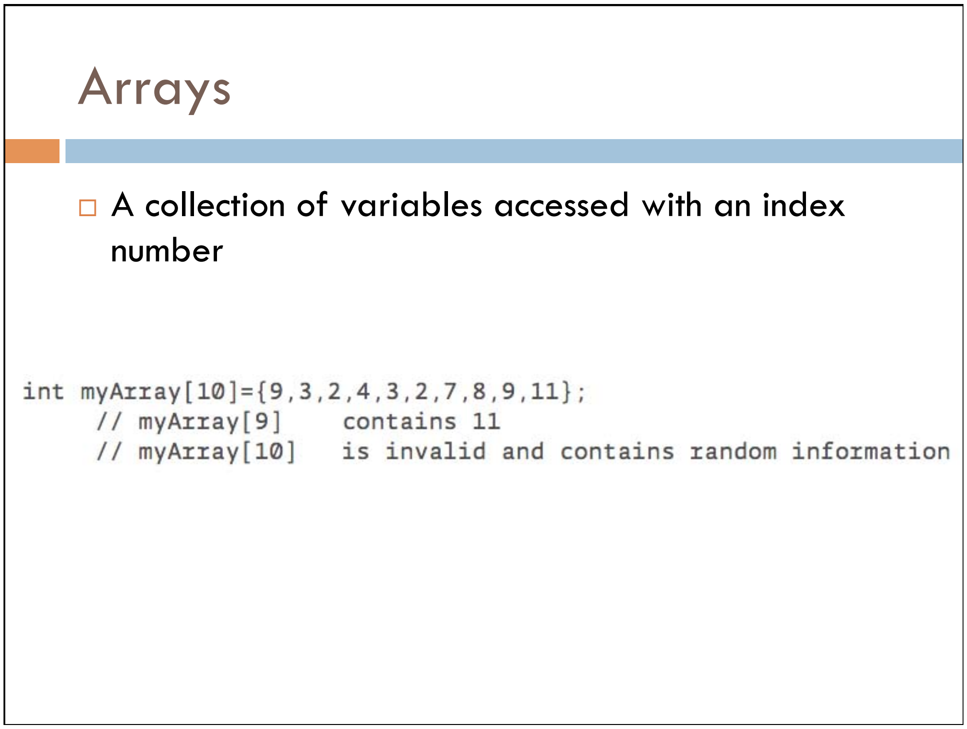
Arrays

- □ A collection of variables accessed with an index number



All of the methods below are valid ways to create (declare) an array.

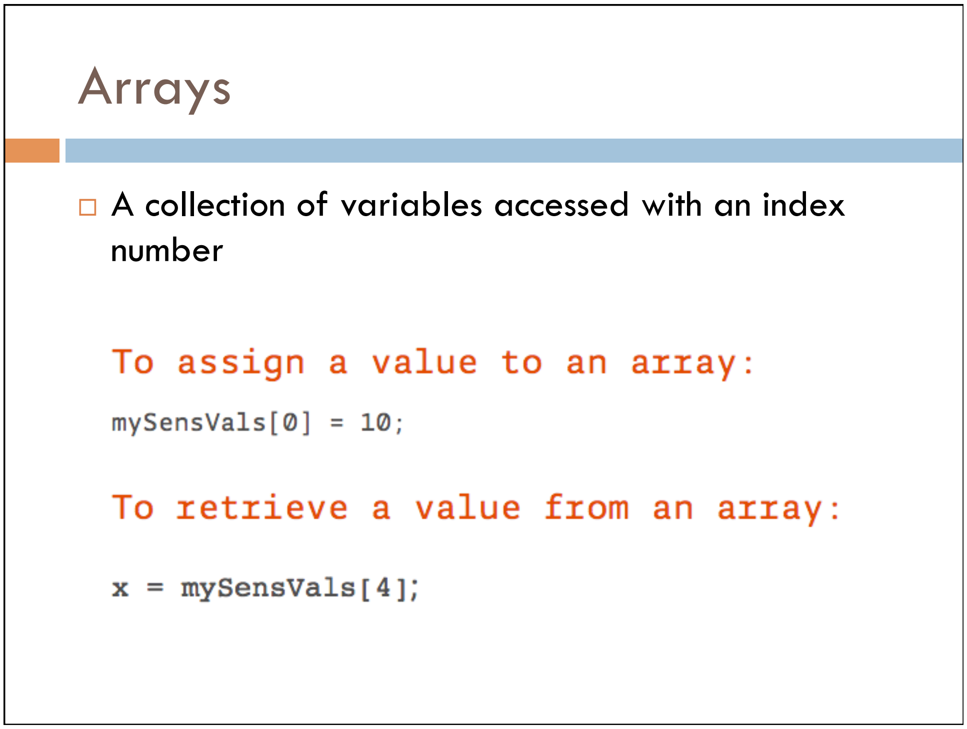
```
int myInts[6];
int myPins[] = {2, 4, 8, 3, 6};
int mySensVals[6] = {2, 4, -8, 3, 2};
char message[6] = "hello";
```



Arrays

- A collection of variables accessed with an index number

```
int myArray[10]={9,3,2,4,3,2,7,8,9,11};
     // myArray[9]    contains 11
     // myArray[10]   is invalid and contains random information
```



Arrays

- A collection of variables accessed with an index number

To assign a value to an array:

```
mySensVals[0] = 10;
```

To retrieve a value from an array:

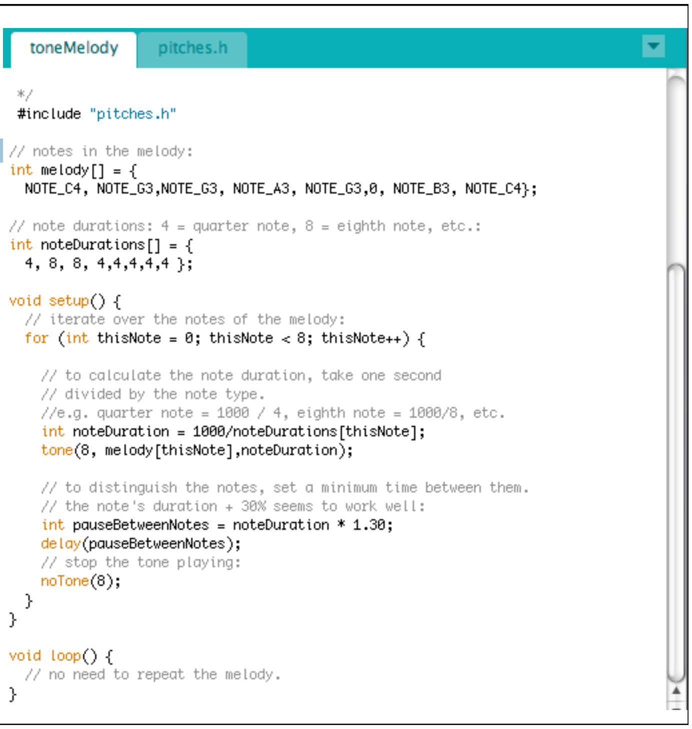
```
x = mySensVals[4];
```

Arrays

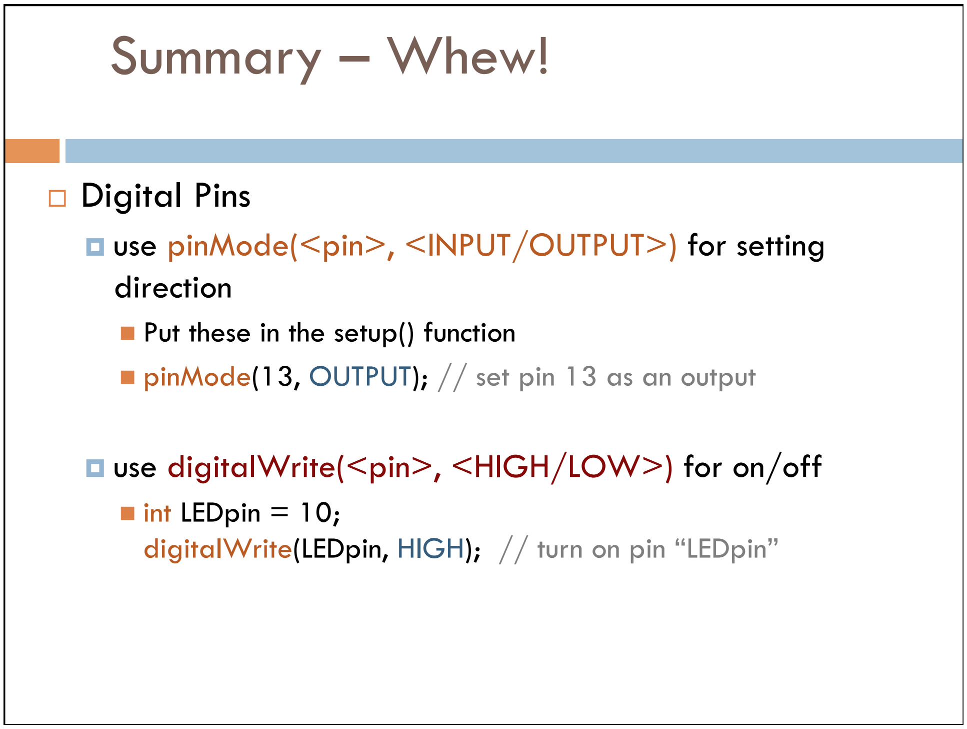
- □ Arrays are often used inside loops…

```
int i;
for (i = 0; i < 5; i = i + 1) {
  Serial.println(myPins[i]);
}
```

Example from Arduino

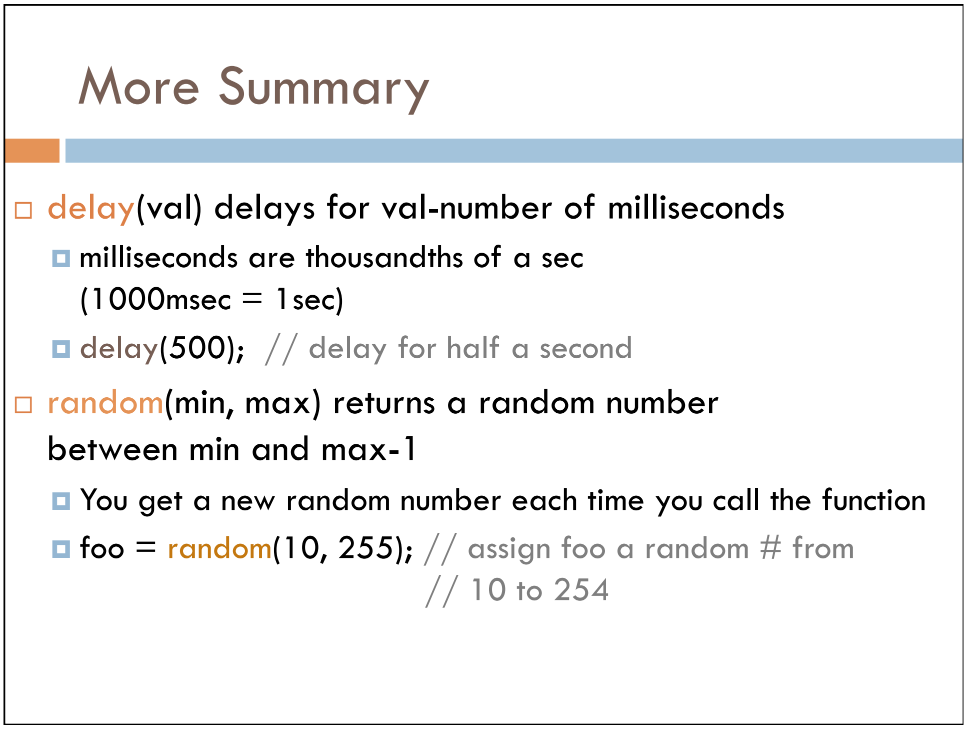


```
 */
 #include "pitches.h"

// notes in the melody:
int melody[] = {
  NOTE_C4, NOTE_G3,NOTE_G3, NOTE_A3, NOTE_G3,0, NOTE_B3, NOTE_C4};

// note durations: 4 = quarter note, 8 = eighth note, etc.:
int noteDurations[] = {
  4, 8, 8, 4,4,4,4,4 };

void setup() {
  // iterate over the notes of the melody:
  for (int thisNote = 0; thisNote < 8; thisNote++) {

    // to calculate the note duration, take one second
    // divided by the note type.
    //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
    int noteDuration = 1000/noteDurations[thisNote];
    tone(8, melody[thisNote],noteDuration);

    // to distinguish the notes, set a minimum time between them.
    // the note's duration + 30% seems to work well:
    int pauseBetweenNotes = noteDuration * 1.30;
    delay(pauseBetweenNotes);
    // stop the tone playing:
    noTone(8);
  }
}

void loop() {
  // no need to repeat the melody.
}
```



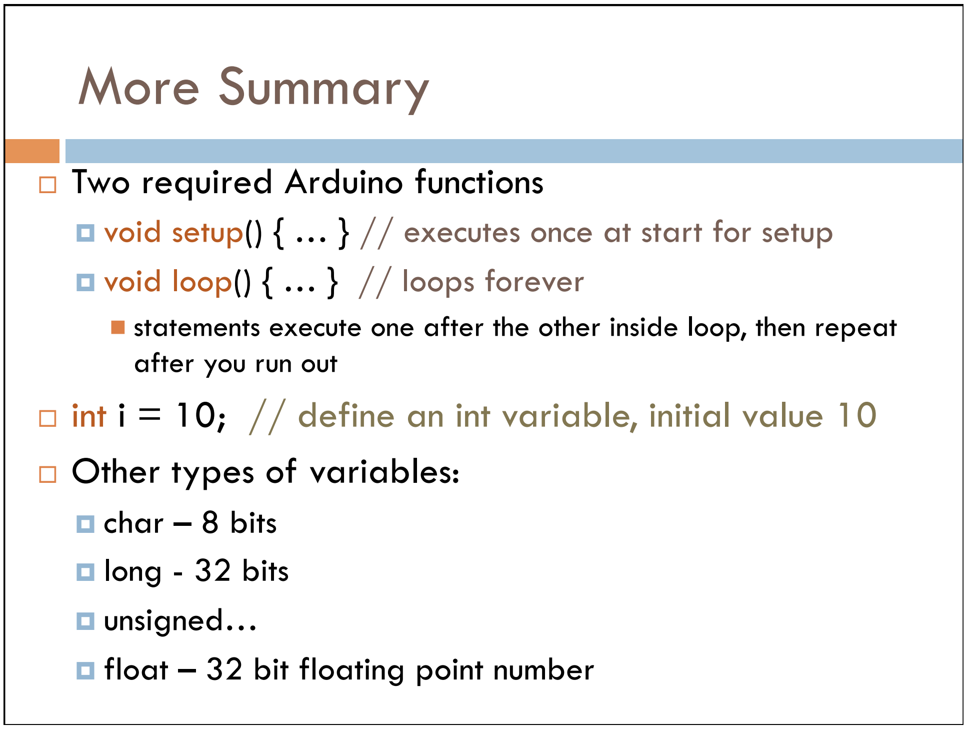
Summary – Whew!

- Digital Pins
  - use pinMode(<pin>, <INPUT/OUTPUT>) for setting direction
    - Put these in the setup() function
    - pinMode(13, OUTPUT); // set pin 13 as an output
  - use digitalWrite(<pin>, <HIGH/LOW>) for on/off
    - int LEDpin = 10;
      digitalWrite(LEDpin, HIGH); // turn on pin "LEDpin"



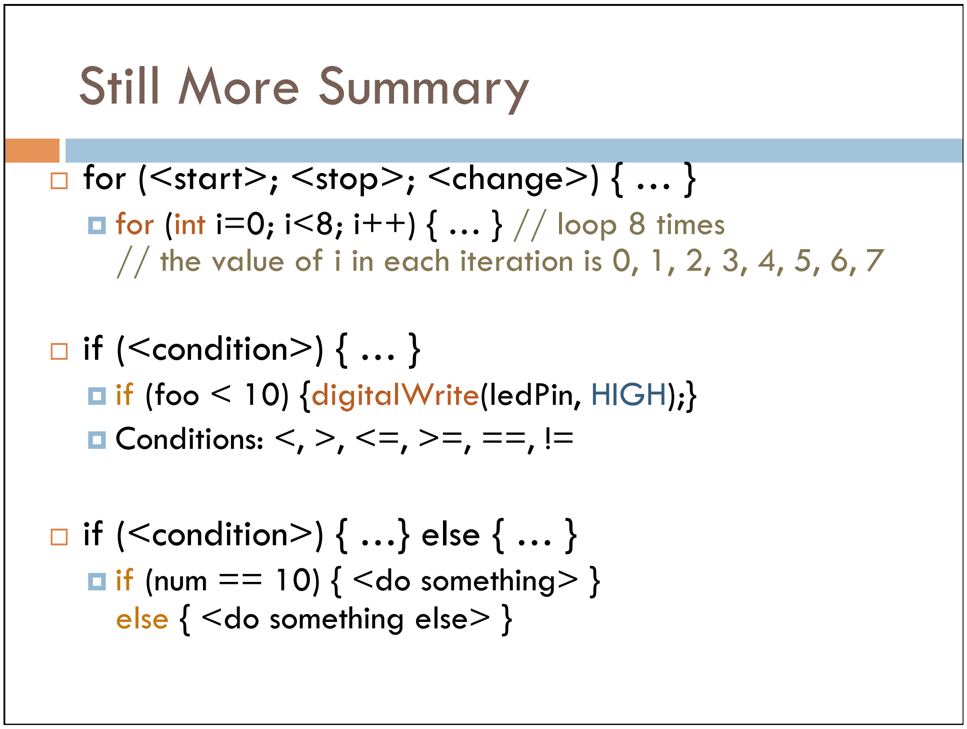
More Summary

- delay(val) delays for val-number of milliseconds
  - milliseconds are thousandths of a sec (1000msec = 1sec)
  - delay(500); // delay for half a second
- random(min, max) returns a random number between min and max-1
  - You get a new random number each time you call the function
  - foo = random(10, 255); // assign foo a random # from
    // 10 to 254



More Summary

- Two required Arduino functions
  - void setup() { … } // executes once at start for setup
  - void loop() { … } // loops forever
    - statements execute one after the other inside loop, then repeat after you run out
- int i = 10; // define an int variable, initial value 10
- Other types of variables:
  - char – 8 bits
  - long - 32 bits
  - unsigned…
  - float – 32 bit floating point number



Still More Summary

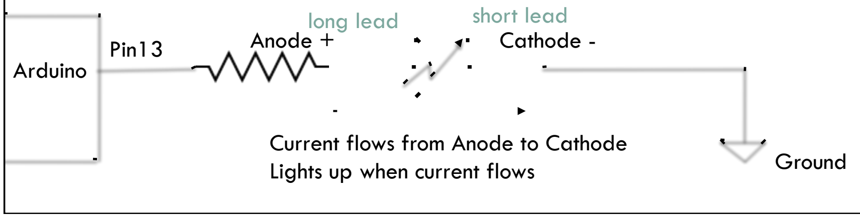
- for (<start>; <stop>; <change>) { … }
  - for (int i=0; i<8; i++) { … } // loop 8 times
    // the value of i in each iteration is 0, 1, 2, 3, 4, 5, 6, 7
- if (<condition>) { … }
  - if (foo < 10) {digitalWrite(ledPin, HIGH);}
  - Conditions: <, >, <=, >=, ==, !=
- if (<condition>) { …} else { … }
  - if (num == 10) { <do something> }
    else { <do something else> }

Speaker

- If you're going to use a speaker, use a current-limiting resistor
  - Most speakers have 8Ω of resistance
  - Some are 4Ω or 16Ω
- Arduino pins can provide or consume 40mA (0.040A)
  - Be conservative – if you're between resistors, use a slightly larger one…
  - 150Ω is a great choice for a speaker

Last Summary (for now)

- LEDs – turn on when current flows from anode to cathode
  - Always use a current-limiting resistor!
  - Remember your resistor color codes
  - 220-470 ohm are good, general-purpose values for LEDs
  - Drive from Arduino on digital pins
  - Use PWM pins if you want to use analogWrite for dimming



Arduino

Pin13

long lead

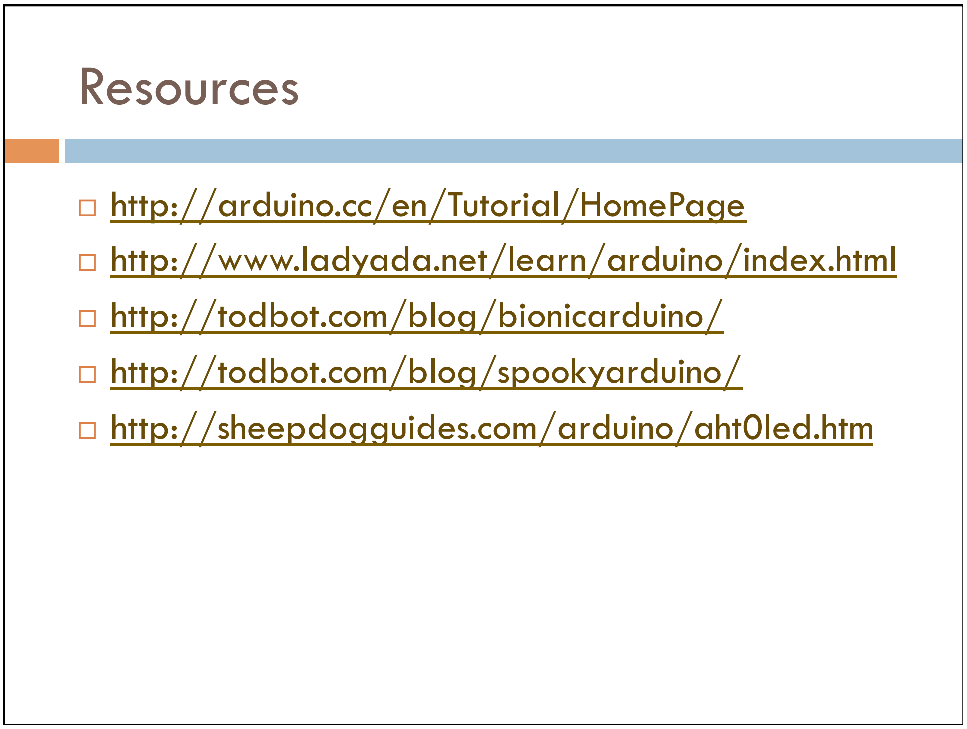
Anode +

short lead

Cathode -

Ground

Current flows from Anode to Cathode
Lights up when current flows



Resources

- □ http://arduino.cc/en/Tutorial/HomePage
- □ http://www.ladyada.net/learn/arduino/index.html
- □ http://todbot.com/blog/bionicarduino/
- □ http://todbot.com/blog/spookyarduino/
- □ http://sheepdogguides.com/arduino/aht0led.htm