Getting Started with Arduino, Sensors, and Rapid System Prototyping

Eric Burger
Fall 2015

Image Source: http://commons.wikimedia.org/wiki/File:Arduino_Logo.svg

What is Arduino?

- Basic Definition: Arduino is an open-source physical computing platform, a collection of simple user-friendly microcontroller boards, and a development environment for writing software. (http://arduino.cc/en/Guide/Introduction)

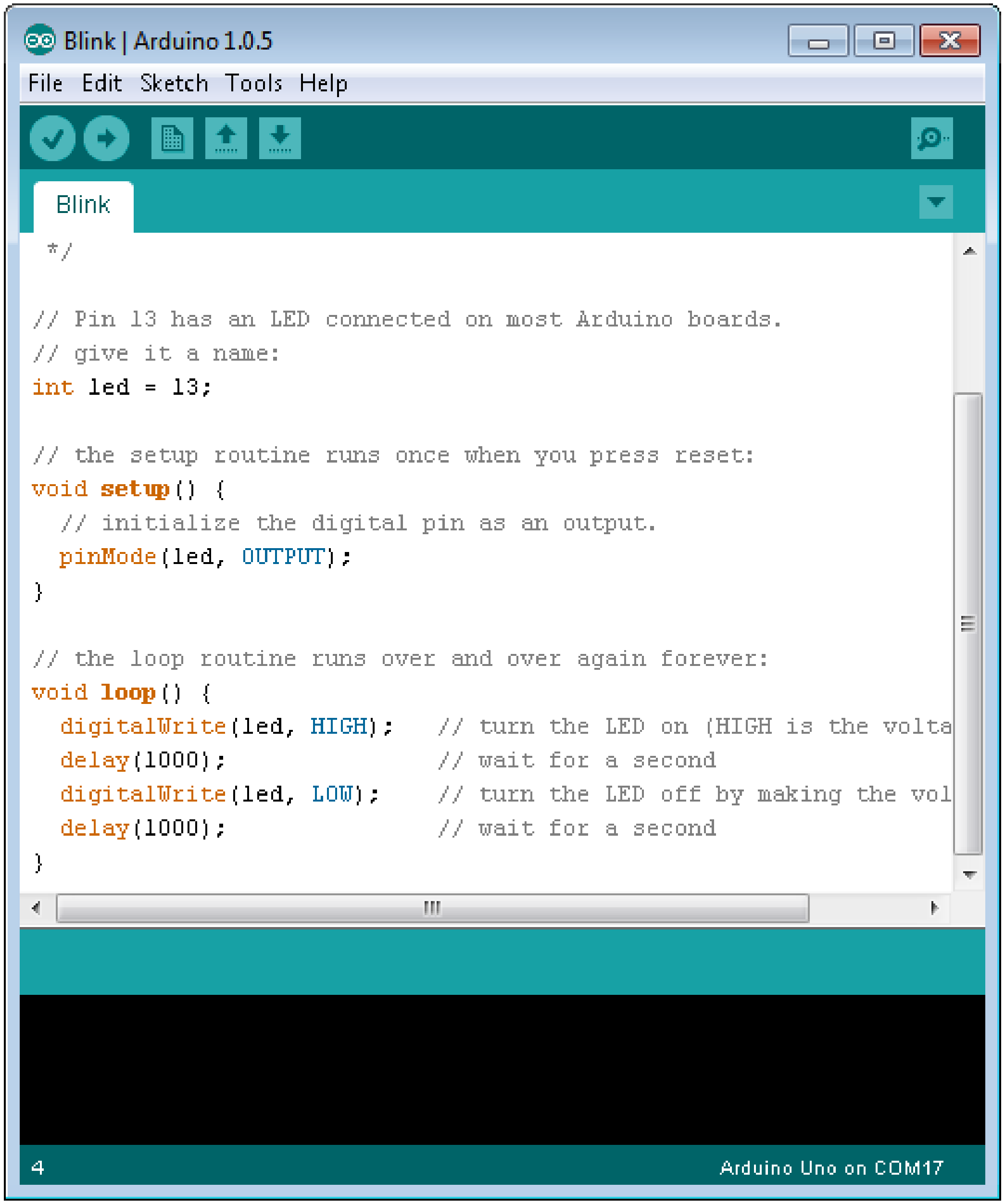
What is Arduino?

- More Relevantly: Arduino is an inexpensive educational tool and prototyping platform supported by a community of manufactures, programmers, students, engineers, designers, and hobbyists.

- No Electronics Experience Required: Arduino makes it possible to focus on the application rather than getting bogged down in memory management and code.

A Closer Look

- Arduino Reference Page
- Program Structure: setup() & loop()
  - Required in every program
- Use “//” or “/* … */” to insert comments
- Color Code:
  - Black = user-defined
  - Blue = constant, type
  - Orange = function or object built into Arduino language

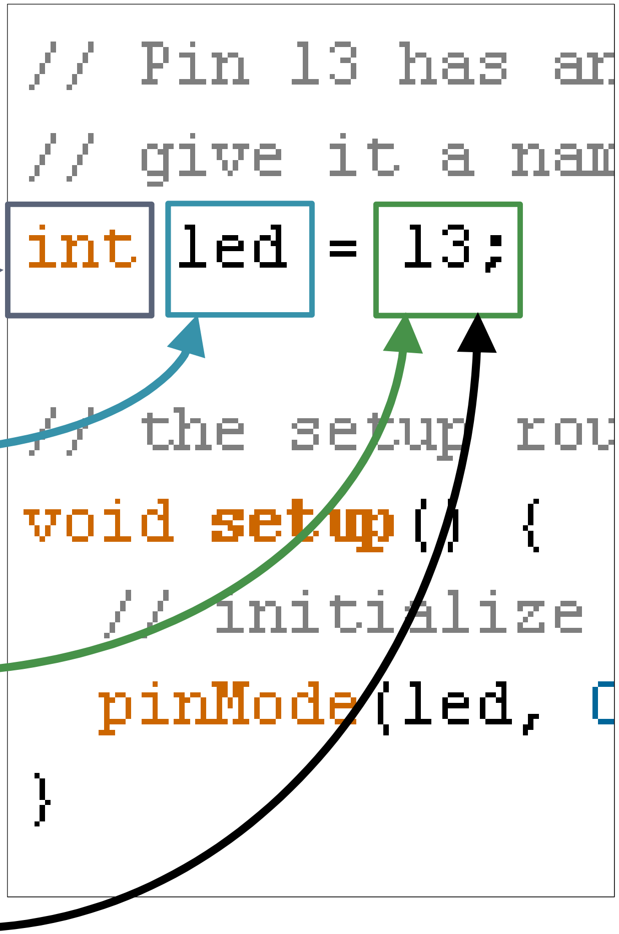


```
 */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);   // turn the LED on (HIGH is the volta
  delay(1000);               // wait for a second
  digitalWrite(led, LOW);    // turn the LED off by making the vol
  delay(1000);               // wait for a second
}
```

An Even Closer Look: Declaring Variables

Variables: For temporarily storing and manipulating data

- Data Types: void, int, char, string, boolean, … see Reference Page
- User-defined name:
  - Give the variable a descriptive name. Should begin with lower-case letter.
- Initialize (optional):
  - You are allowed to create a variable without giving it an initial value, but it is a good idea to initialize.
- Syntax: Use "=" to assign values and end every line with ";"



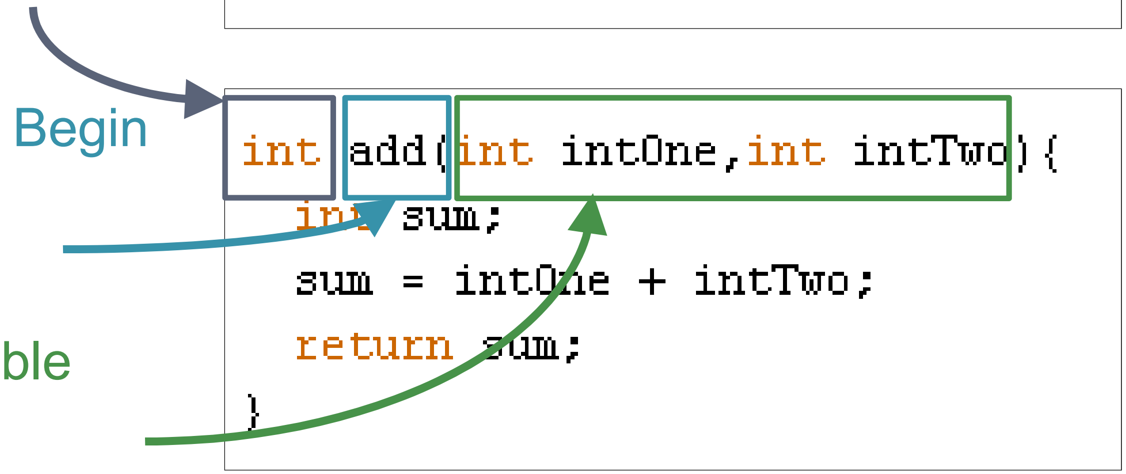
```
// Pin 13 has an
// give it a nam
int led = 13;

// the setup rou
void setup() {
  // initialize
  pinMode(led, O
}
```

An Even Closer Look: Defining Functions

Functions: Accept input, perform actions, return data

- Return Type:
  - Type of data returned by function.
- User-defined name
  - Give a descriptive name. Begin with lower-case letter.
- Inputs (optional)
  - List input types and variable names within “(   )”
  - Input variables only exist within function.

```
int add(int intOne,int intTwo){
  int sum;
  sum = intOne + intTwo;
  return sum;
}
```

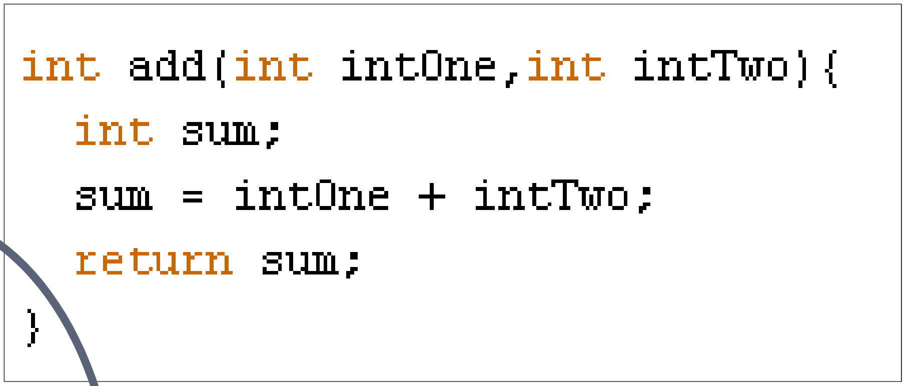


```
int add(int intOne,int intTwo){
  int sum;
  sum = intOne + intTwo;
  return sum;
}
```

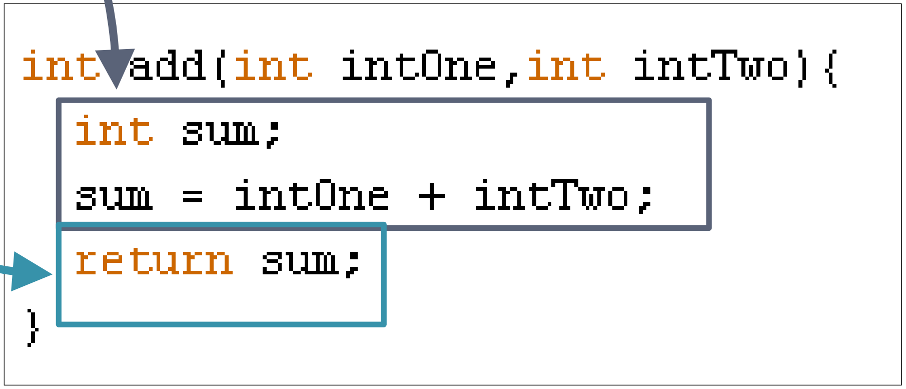
```
void loop(){
  int result = add(1,2);
}
```

An Even Closer Look: Creating Functions

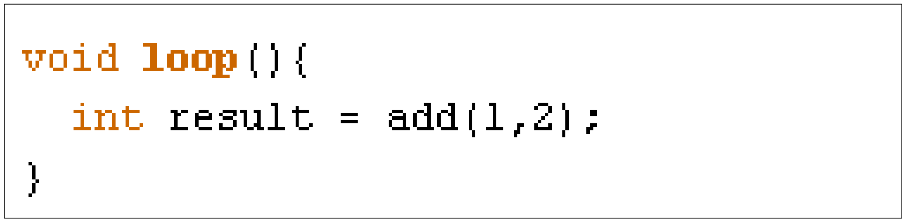
- Perform Action:
  - Input variables were “passed” to the function and have already been declared and initialized.
- Return:
  - Return data that matches the specified “Return Type”
- Syntax:
  - Name followed by “( )”
  - Actions performed by function are between “{   }”
  - “return” followed by variable.
  - Any code after the return statement will not be executed.



```
int add(int intOne,int intTwo){
  int sum;
  sum = intOne + intTwo;
  return sum;
}
```



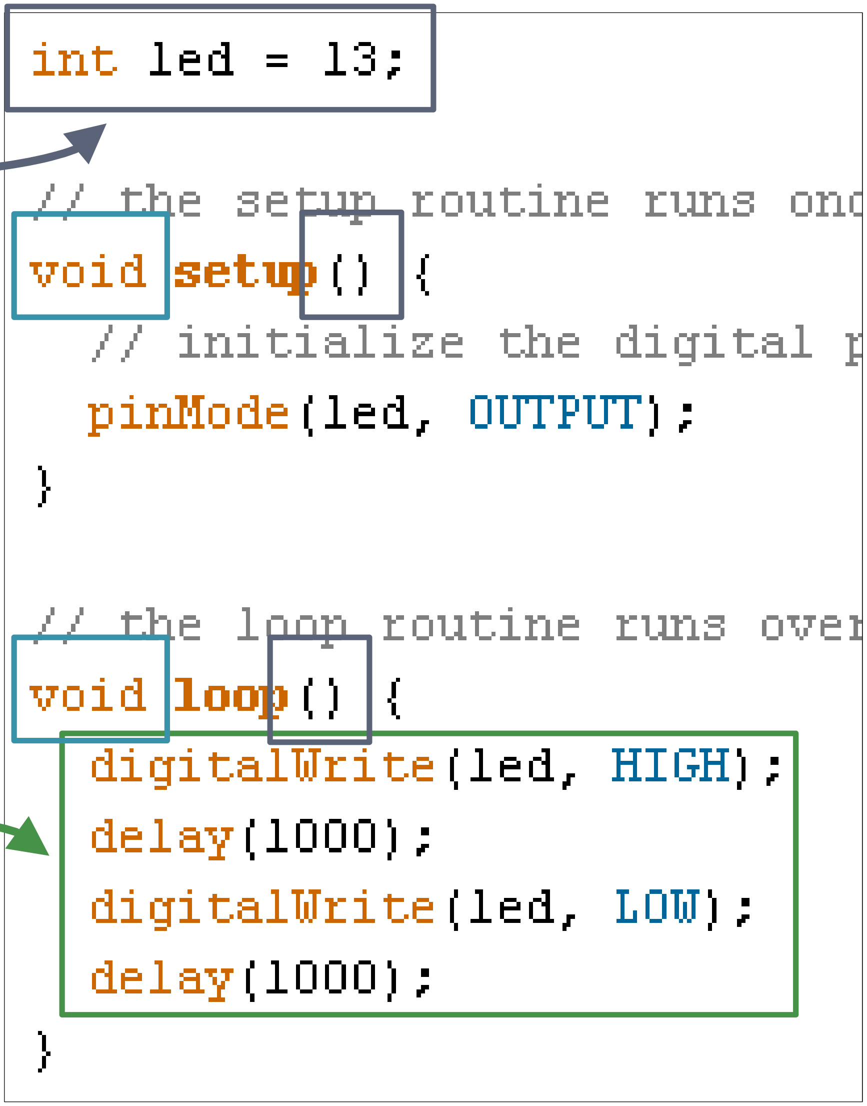
```
int add(int intOne,int intTwo){
  int sum;
  sum = intOne + intTwo;
  return sum;
}
```



```
void loop(){
  int result = add(1,2);
}
```

Back to Blink Program

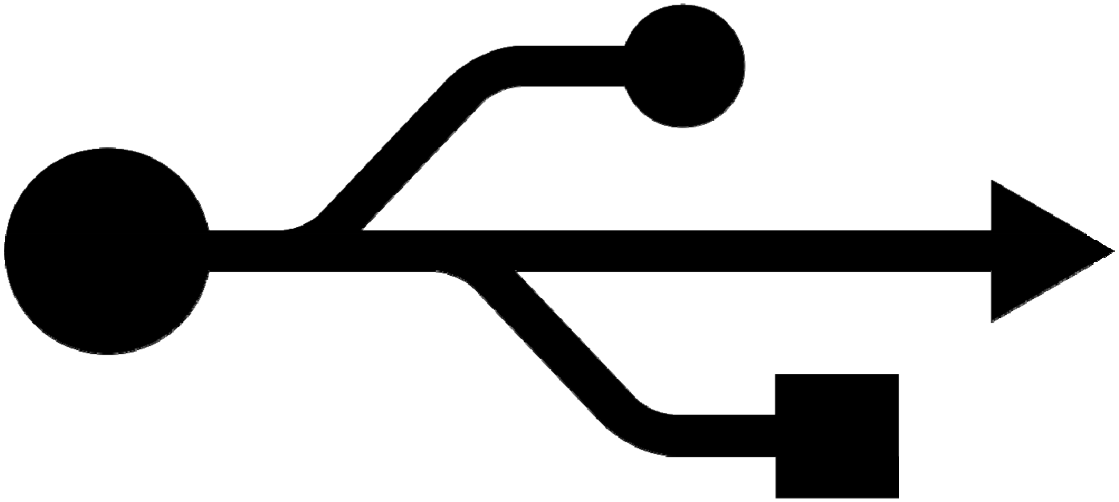
- Variables: If declared before setup() and loop() then the variable can be used anywhere in the program.
- Type:
  - Set to "void" for setup() and loop()
  - When return type is void, "return;" line is optional.
- Inputs: None
- Function calls:
  - loop() performs 4 built-in functions
  - Does not return anything (void)



```
int led = 13;

// the setup routine runs onc
void setup() {
  // initialize the digital p
  pinMode(led, OUTPUT);
}

// the loop routine runs over
void loop() {
  digitalWrite(led, HIGH);
  delay(1000);
  digitalWrite(led, LOW);
  delay(1000);
}
```

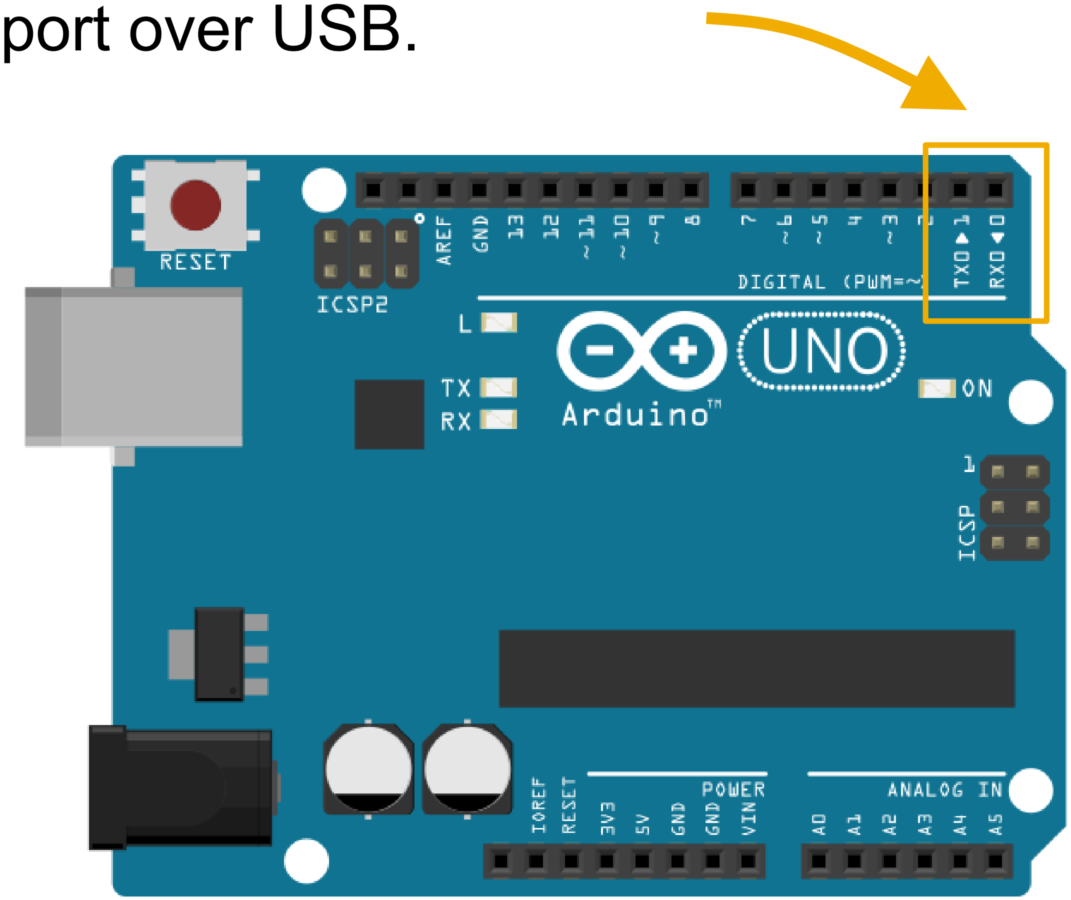
Serial Communication

- Basic Definition: Send data one bit or byte at a time over a communication channel (as opposed to several bytes sent as a whole).



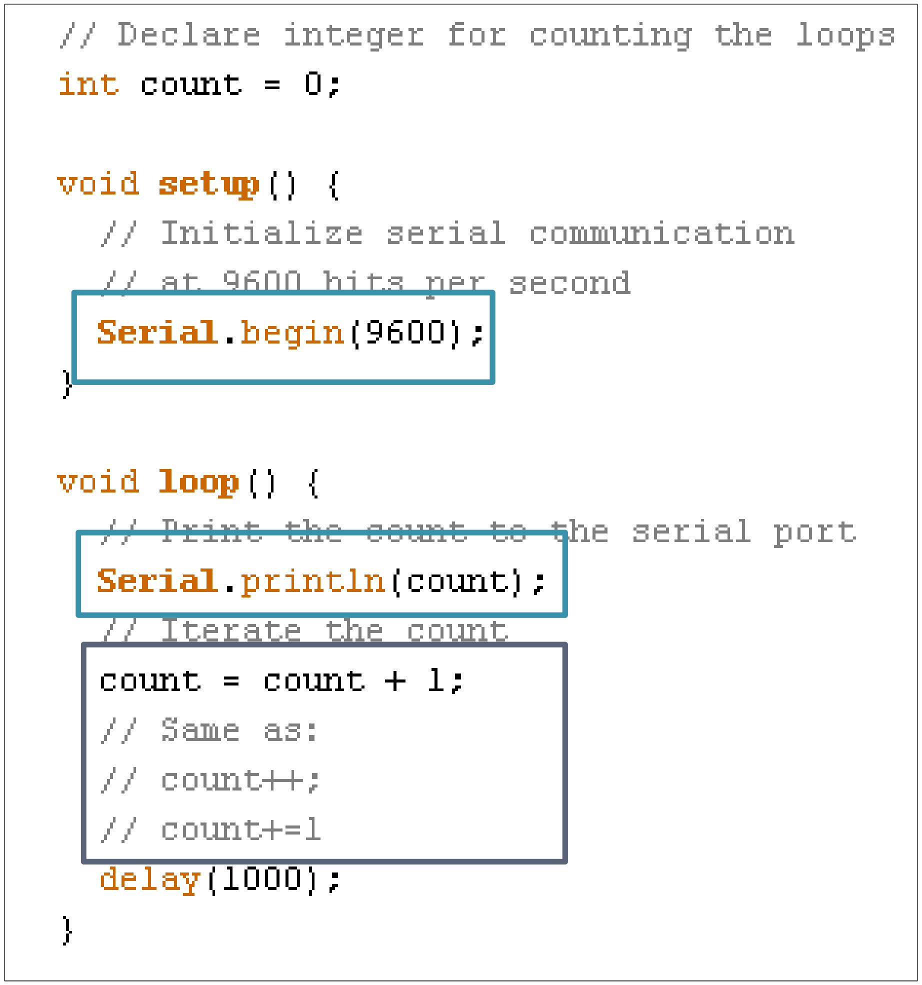
Serial Communication

- The Arduino Uno includes a build-in “hardware” serial port (Digital Pins 0 and 1) and electronics for communicating with this port over USB.



Listening to the Arduino

- Serial Object: Packages variables and functions
  - Makes it easier to write code
  - “Object”+ “.”+” “Function”
  - Arduino sends the variables value to your computer as a character string
- Syntax:
  - count = count + 1 assigns a new value to count variable
  - count++ increases count by 1
  - count += 2 increases count by 2

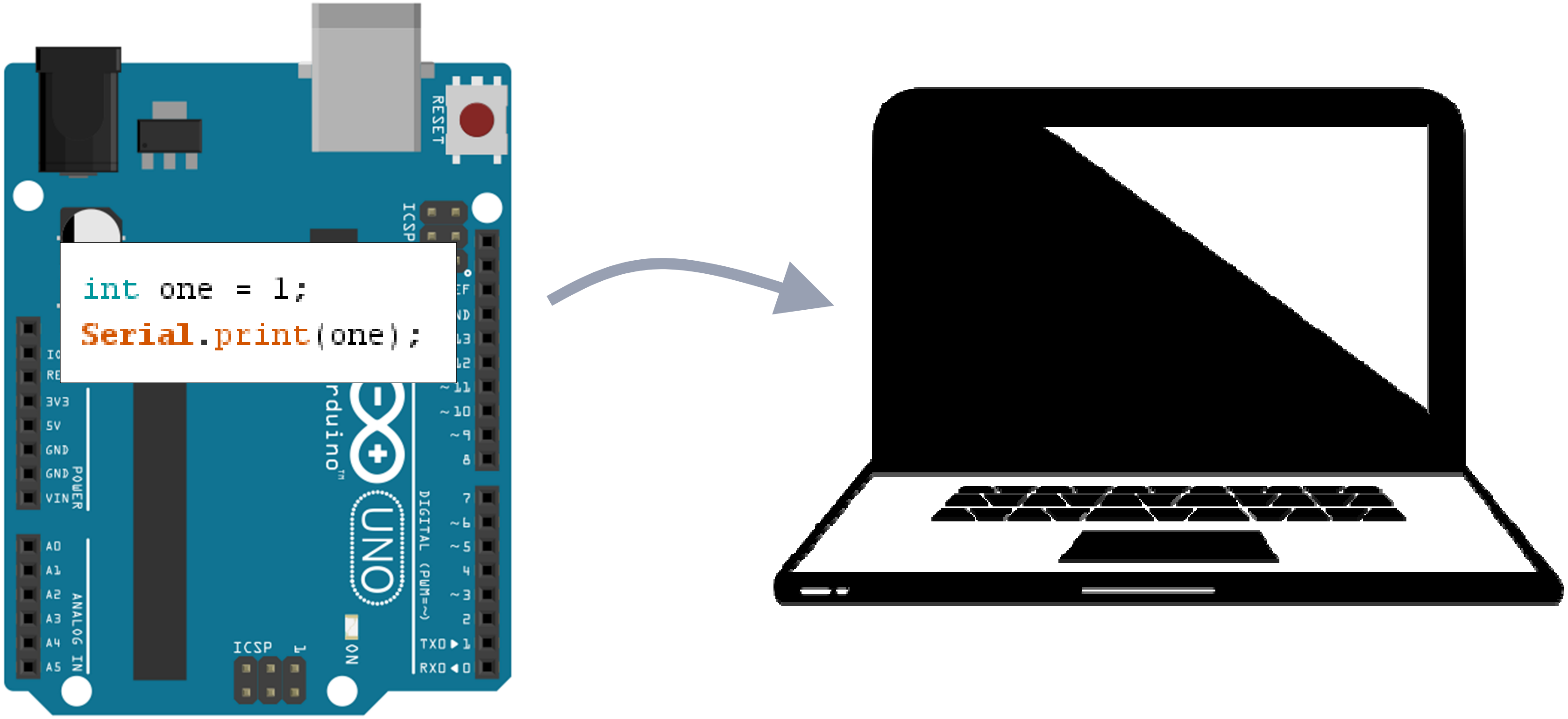


```
// Declare integer for counting the loops
int count = 0;

void setup() {
  // Initialize serial communication
  // at 9600 bits per second
  Serial.begin(9600);
}

void loop() {
  // Print the count to the serial port
  Serial.println(count);
  // Iterate the count
  count = count + 1;
  // Same as:
  // count++;
  // count+=1
  delay(1000);
}
```

Aside: Serial.print(…)

If you “print” the integer 1 …

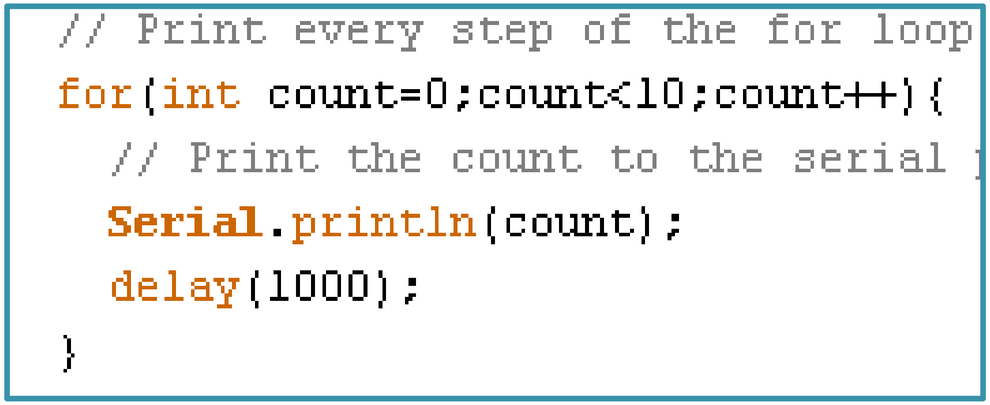
… your computer receives the character ‘1’.



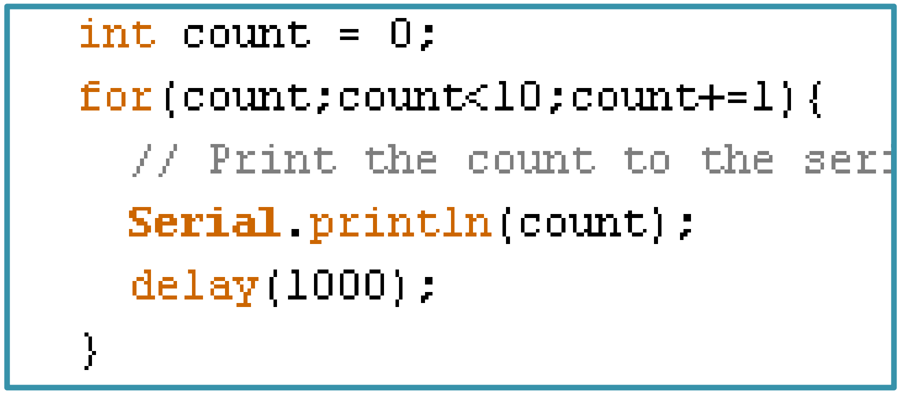
```
int one = 1;
Serial.print(one);
```

Creating a “For Loop”

- For Loop:
  - 1st: Variable to be used
  - 2nd: Check if loop is finished
  - 3rd: Increment the variable
- Syntax: Separate with “;”
- Best Practice: Declare the counting variable in the function call.



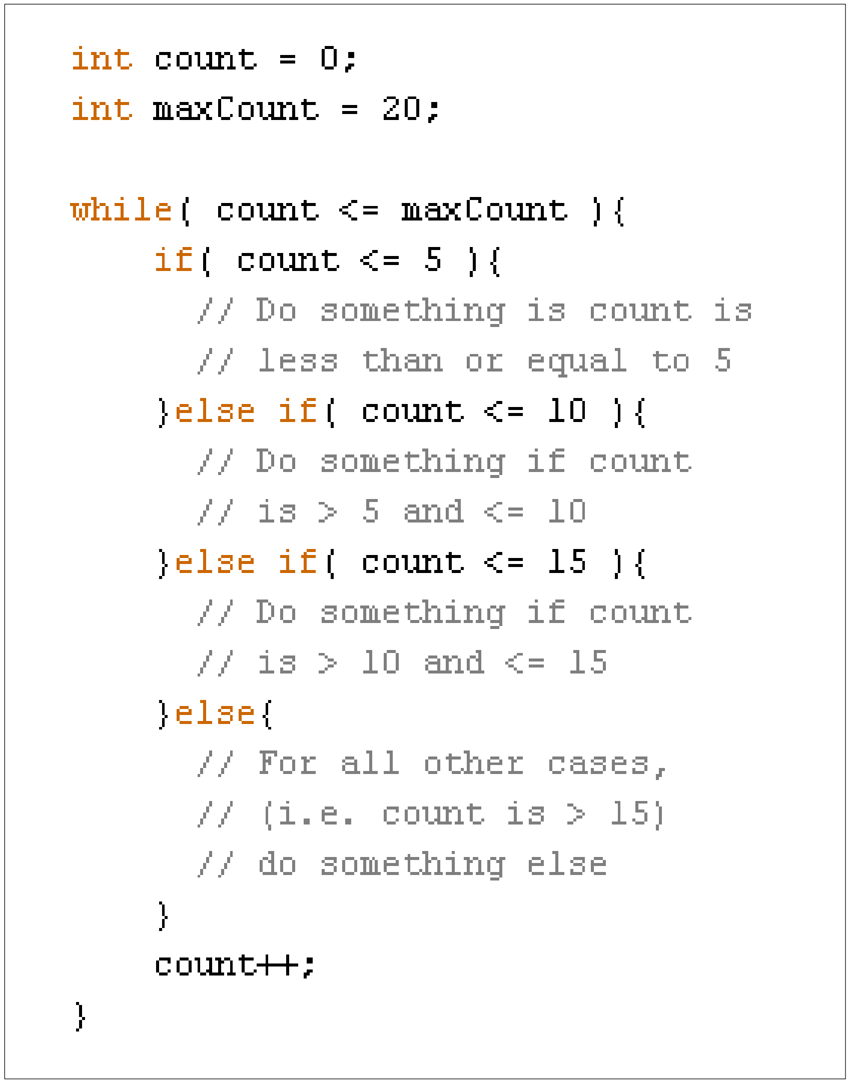
```
// Print every step of the for loop
for(int count=0;count<10;count++){
  // Print the count to the serial
  Serial.println(count);
  delay(1000);
}
```



```
 int count = 0;
 for(count;count<10;count+=1){
   // Print the count to the ser
   Serial.println(count);
   delay(1000);
 }
```

While…If…Else if…Else

- While Loop: Continue loop as long as Boolean expression is true
- If Statement: Enter only if Boolean is true
- Else If Statement: If previous statement was false, check Boolean
- Else Statement: Enter if all previous statements were false



```
int count = 0;
int maxCount = 20;

while( count <= maxCount ){
    if( count <= 5 ){
      // Do something is count is
      // less than or equal to 5
    }else if( count <= 10 ){
      // Do something if count
      // is > 5 and <= 10
    }else if( count <= 15 ){
      // Do something if count
      // is > 10 and <= 15
    }else{
      // For all other cases,
      // (i.e. count is > 15)
      // do something else
    }
    count++;
}
```

Learning Resources

- Arduino IDE Examples
- Arduino.cc website
- Sparkfun.com and Adafruit.com tutorials