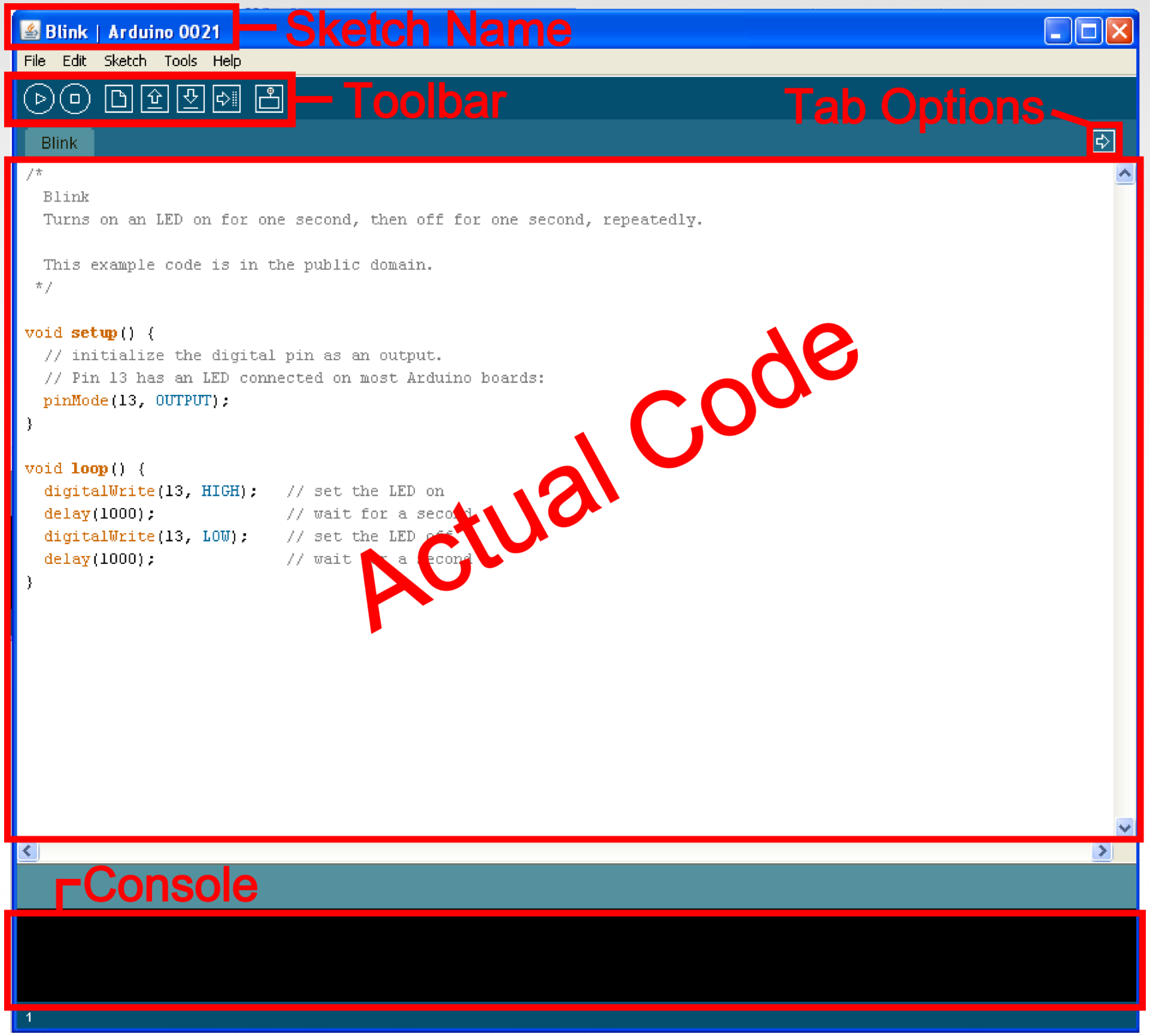
Code

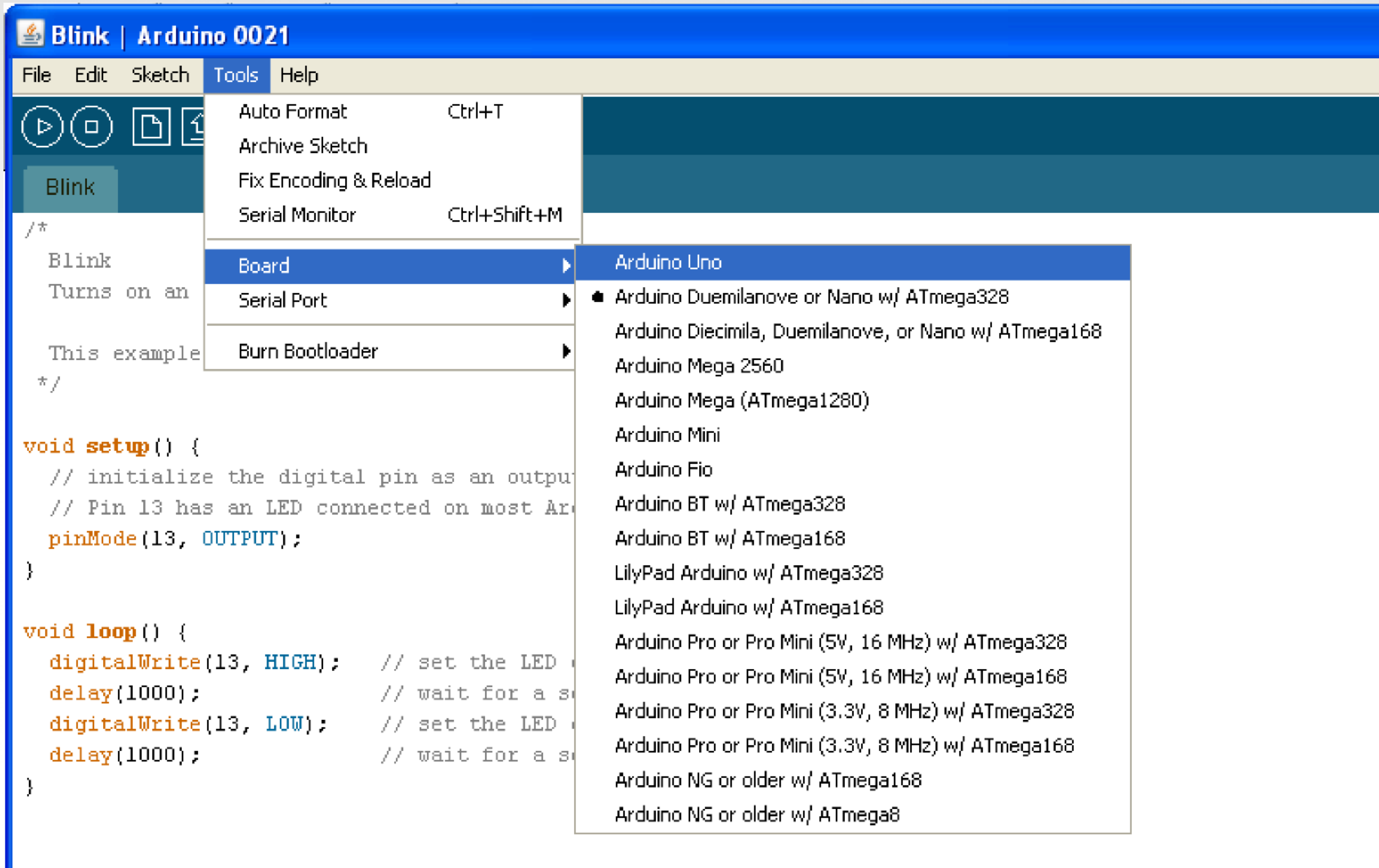
http://arduino.cc/en/Reference/HomePage

The Arduino Environment



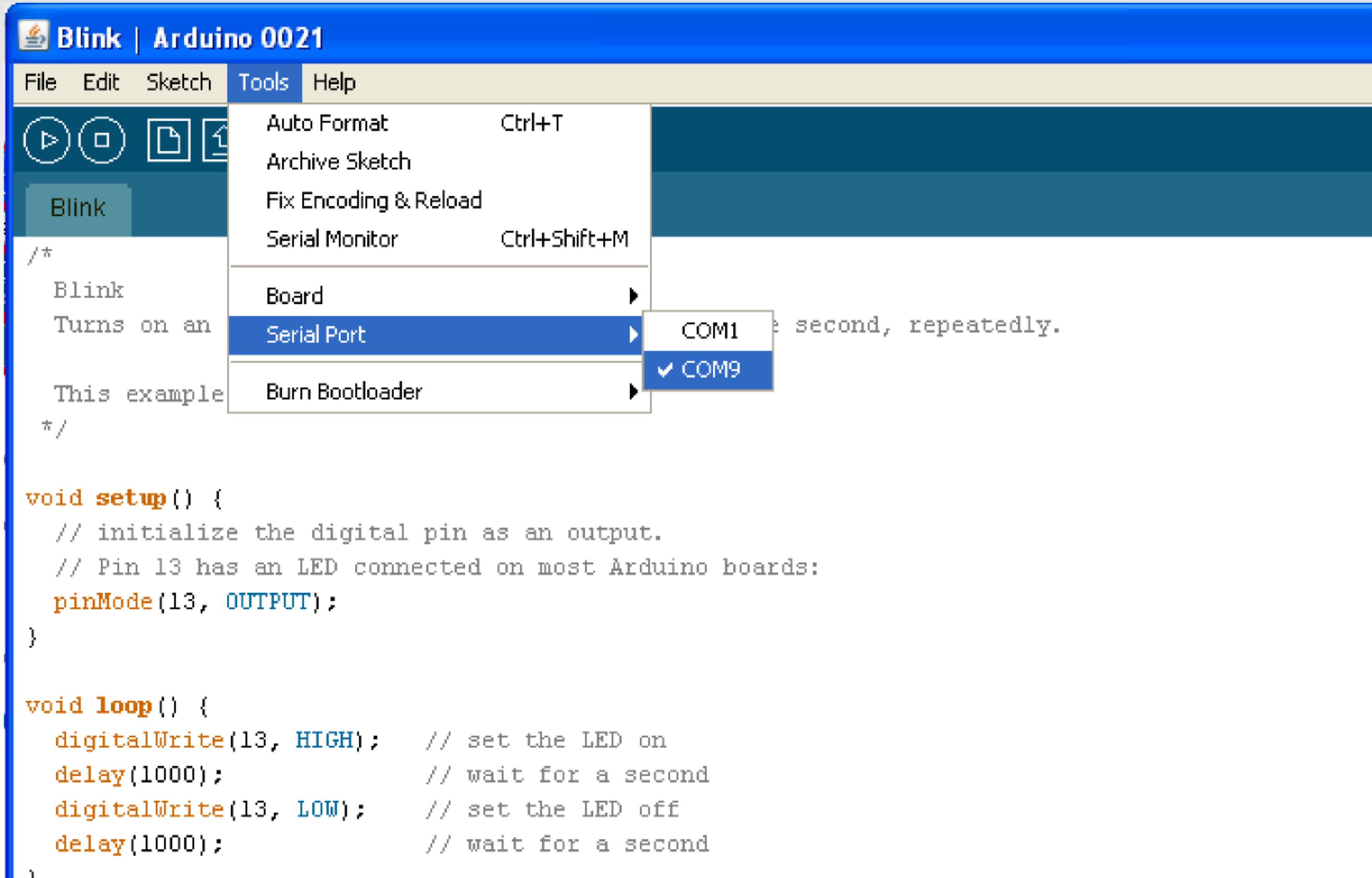
```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  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
}
```

Board Type


Blink | Arduino 0021
File
Edit
Sketch
Tools
Help
Blink
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Board
Serial Port
Burn Bootloader
Arduino Uno
● Arduino Duemilanove or Nano w/ ATmega328
Arduino Diecimila, Duemilanove, or Nano w/ ATmega168
Arduino Mega 2560
Arduino Mega (ATmega1280)
Arduino Mini
Arduino Fio
Arduino BT w/ ATmega328
Arduino BT w/ ATmega168
LilyPad Arduino w/ ATmega328
LilyPad Arduino w/ ATmega168
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega168
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega168
Arduino NG or older w/ ATmega168
Arduino NG or older w/ ATmega8

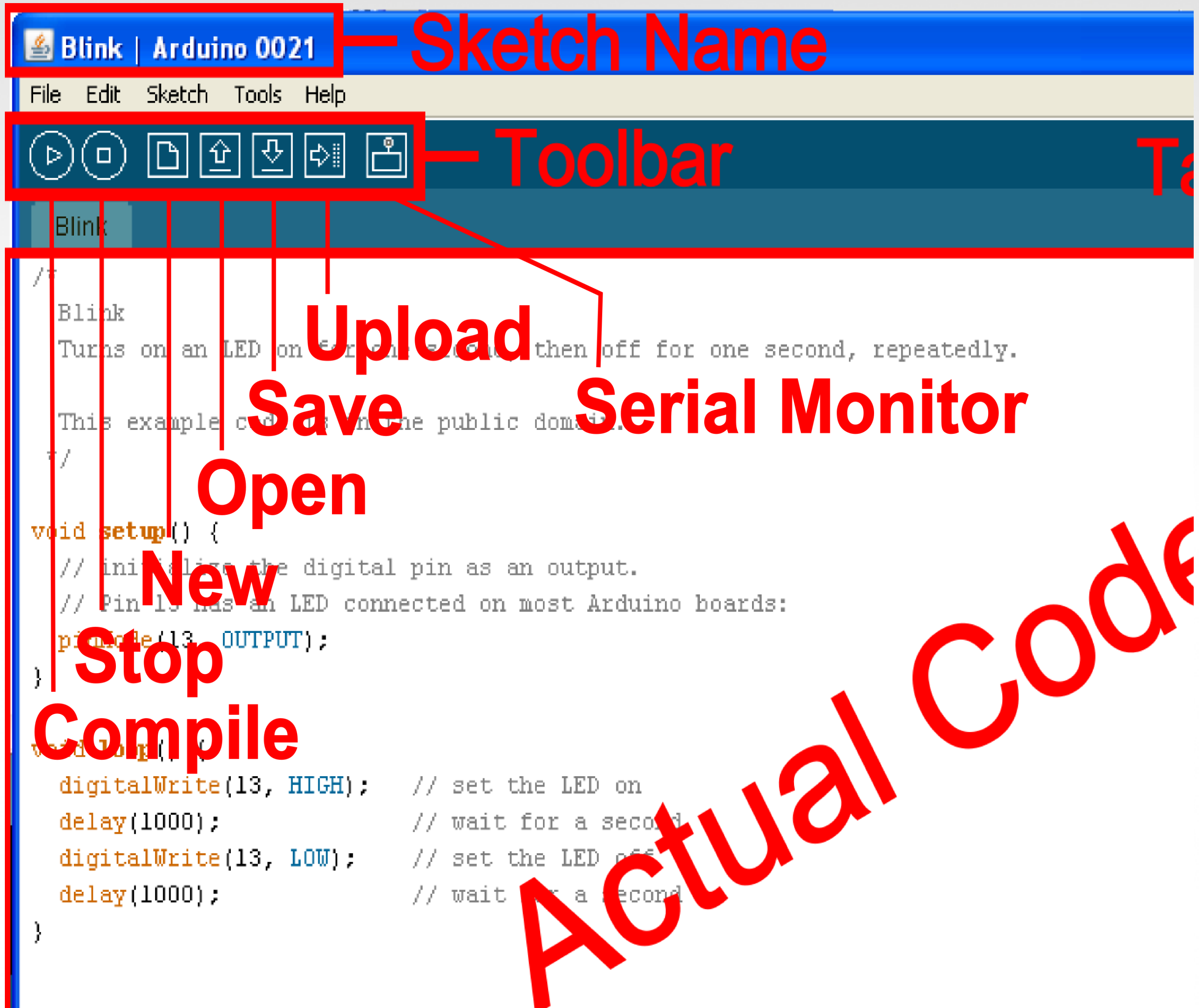
```
/*
  Blink
  Turns on an

  This example
 */

void setup() {
  // initialize the digital pin as an outpu
  // Pin 13 has an LED connected on most Ar
  pinMode(13, OUTPUT);
}

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

Serial Port / COM Port



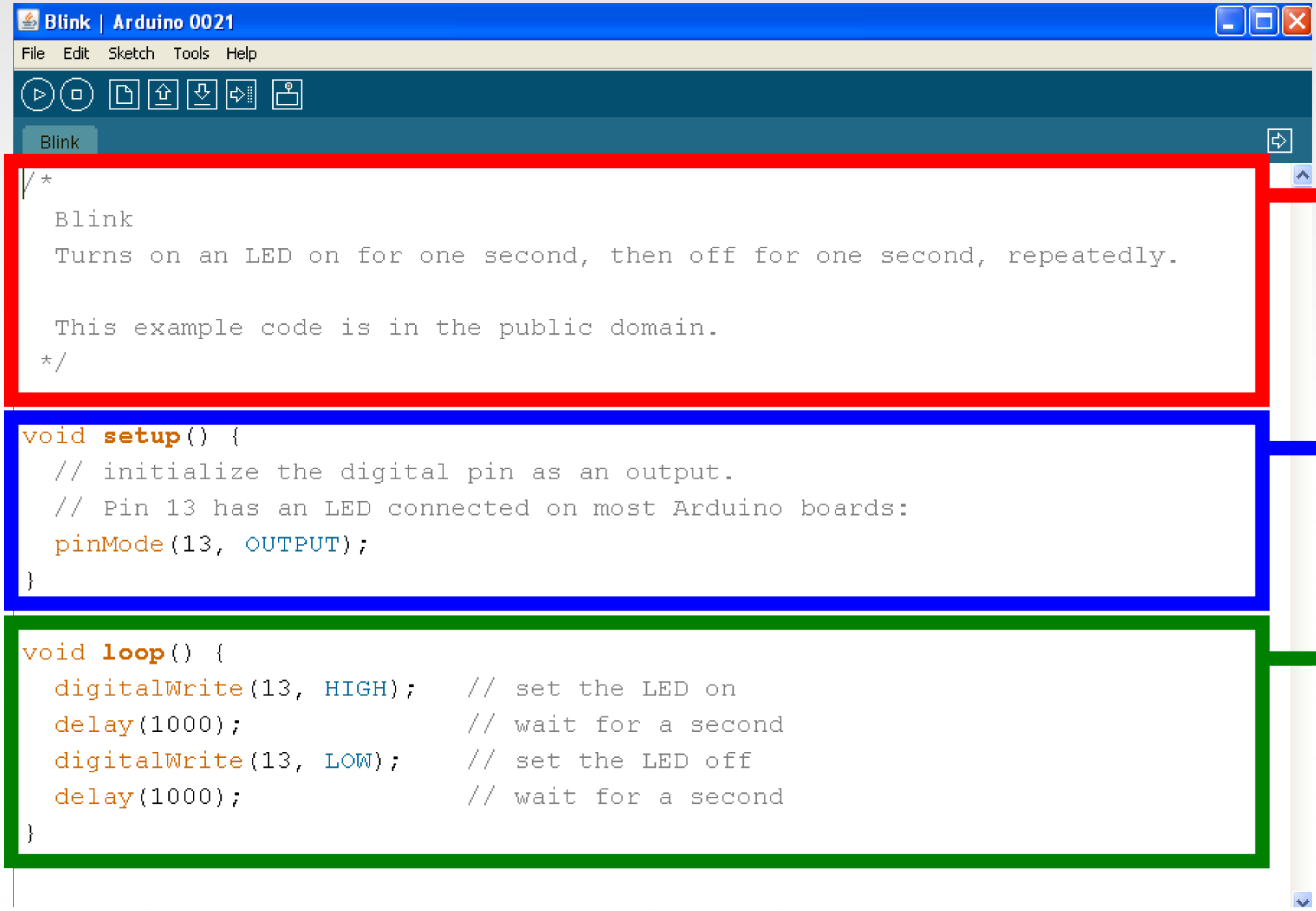
```
/*
  Blink
  Turns on an                                   second, repeatedly.

  This example
 */

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
```

The Environment



```
/*
  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
}
```

Parts of the Sketch



```
/*
  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
}
```

Comments / Explaining the game

Setup / Stretching or tying shoes

Loop / Playing the game

Comments

- Comments can be anywhere

Comments

- Comments can be anywhere
- Comments created with // or /* and */

Comments

- Comments can be anywhere
- Comments created with // or /* and */
- Comments do not affect code

Comments

- Comments can be anywhere
- Comments created with // or /* and */
- Comments do not affect code
- You may not need comments, but think about the community!

Operators

The equals sign

= is used to assign a value

== is used to compare values

Operators

And & Or

&& is “and”

|| is “or”

Variables

Basic variable types:

Boolean

Integer

Character

Declaring Variables

Boolean: ***boolean variableName;***

Declaring Variables

Boolean: ***boolean variableName;***

Integer: ***int variableName;***

Declaring Variables

Boolean: ***boolean variableName;***

Integer: ***int variableName;***

Character: ***char variableName;***

Declaring Variables

Boolean: ***boolean variableName;***

Integer: ***int variableName;***

Character: ***char variableName;***

String: ***char stringName [ ];***

Assigning Variables

Boolean: ***variableName = true;***

or ***variableName = false;***

Assigning Variables

Boolean: ***variableName = true;***

or ***variableName = false;***

Integer: ***variableName = 32767;***

or ***variableName = -32768;***

Assigning Variables

Boolean: ***variableName = true;***

or ***variableName = false;***

Integer: ***variableName = 32767;***

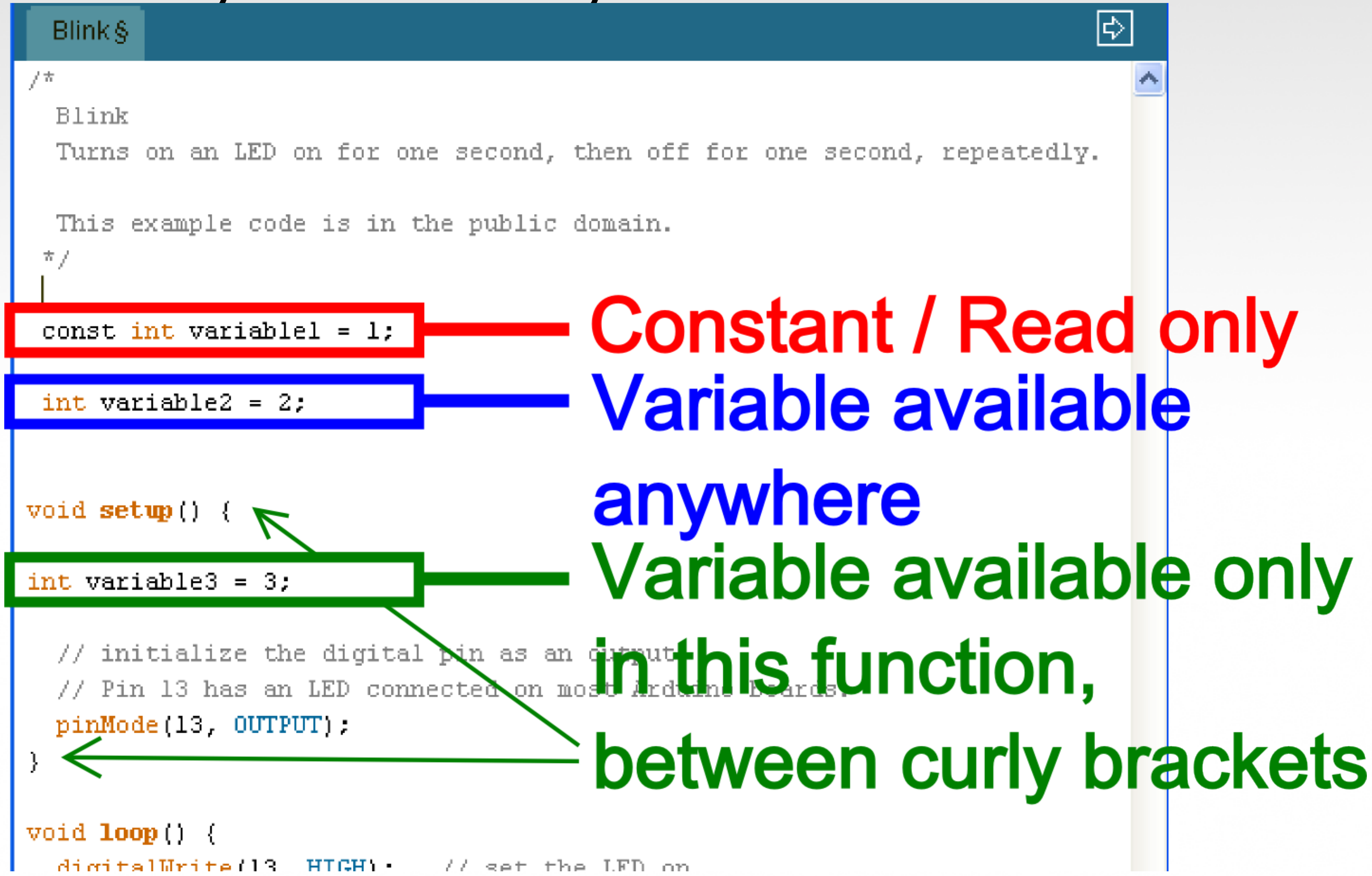
or ***variableName = -32768;***

Character: ***variableName = ‘A’;***

or ***stringName = “SparkFun”;***

Variable Scope

Where you declare your variables matters



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

  This example code is in the public domain.
 */

 const int variable1 = 1;

 int variable2 = 2;


void setup() {

int variable3 = 3;

  // 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
```

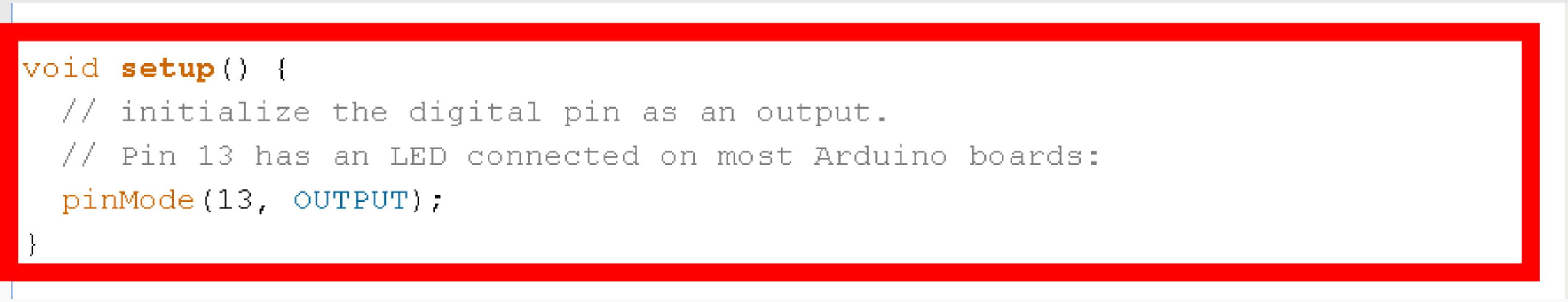
Constant / Read only

Variable available anywhere

Variable available only in this function, between curly brackets

Setup

void setup ( ) { }

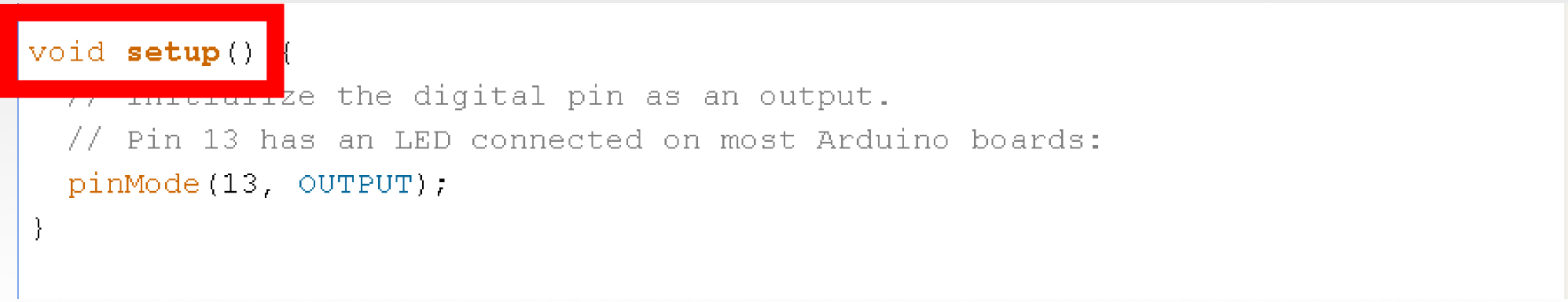


```
void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
}
```

The setup function comes before the loop function and is necessary for all Arduino sketches

Setup

void setup ( ) { }



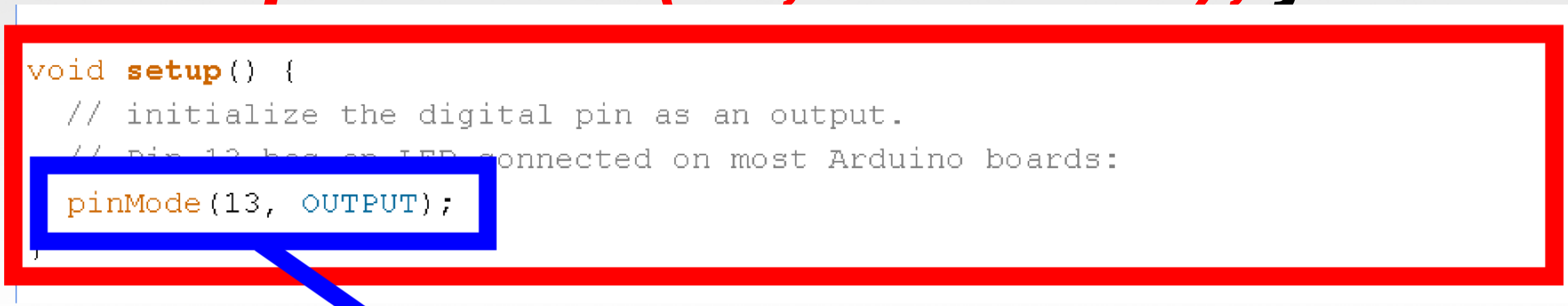
```
void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
}
```

The setup header will never change, everything else that occurs in setup happens inside the curly brackets

Setup

void setup ( ) {

pinMode (13, OUTPUT); }



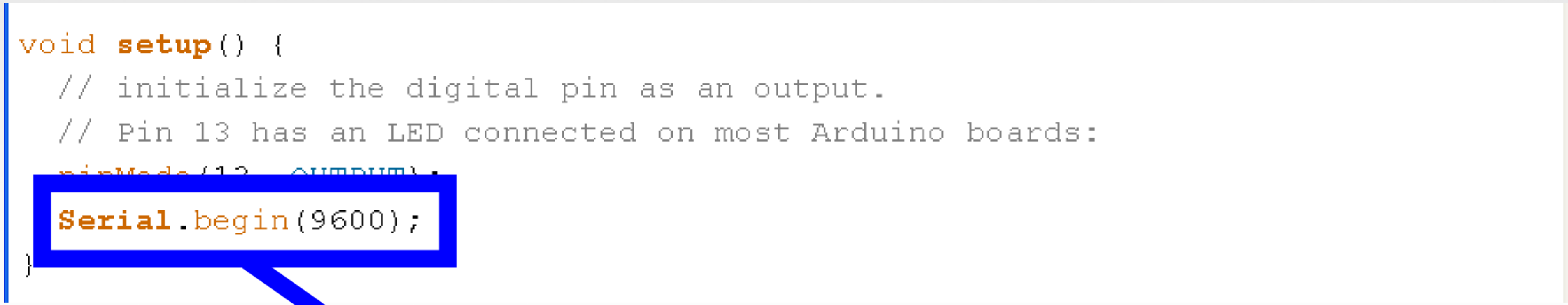
```
void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
}
```

Outputs are declare in setup, this is done by using the pinMode function

This particular example declares digital pin # 13 as an output, remember to use CAPS

Setup

void setup ( ) { Serial.begin;}



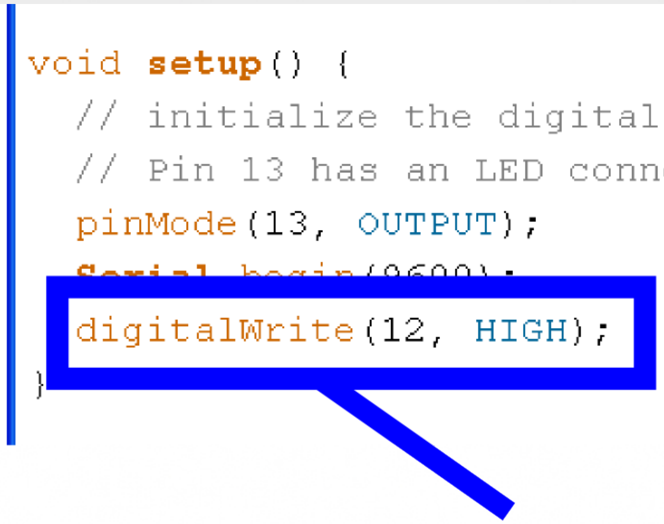
```
void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
  Serial.begin(9600);
}
```

Serial communication also begins in setup

This particular example declares Serial communication at a baud rate of 9600. More on Serial later...

Setup, Internal Pullup Resistors

void setup ( ) {
digitalWrite (12, HIGH); }



```
void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
  Serial.begin(9600);
  digitalWrite(12, HIGH);
}
```

You can also create internal pullup resistors in setup, to do so digitalWrite the pin HIGH

This takes the place of the pullup resistors currently on your circuit 7 buttons

Setup, Interrupts

void setup ( ) {

attachInterrupt (interrupt, function, mode) }

You can designate an interrupt function to Arduino pins # 2 and 3

This is a way around the linear processing of Arduino

Setup, Interrupts

void setup ( ) {

attachInterrupt (interrupt, function, mode) }

Interrupt: the number of the interrupt, 0 or 1, corresponding to Arduino pins # 2 and 3 respectively

Function: the function to call when the interrupt occurs

Mode: defines when the interrupt should be triggered

Setup, Interrupts

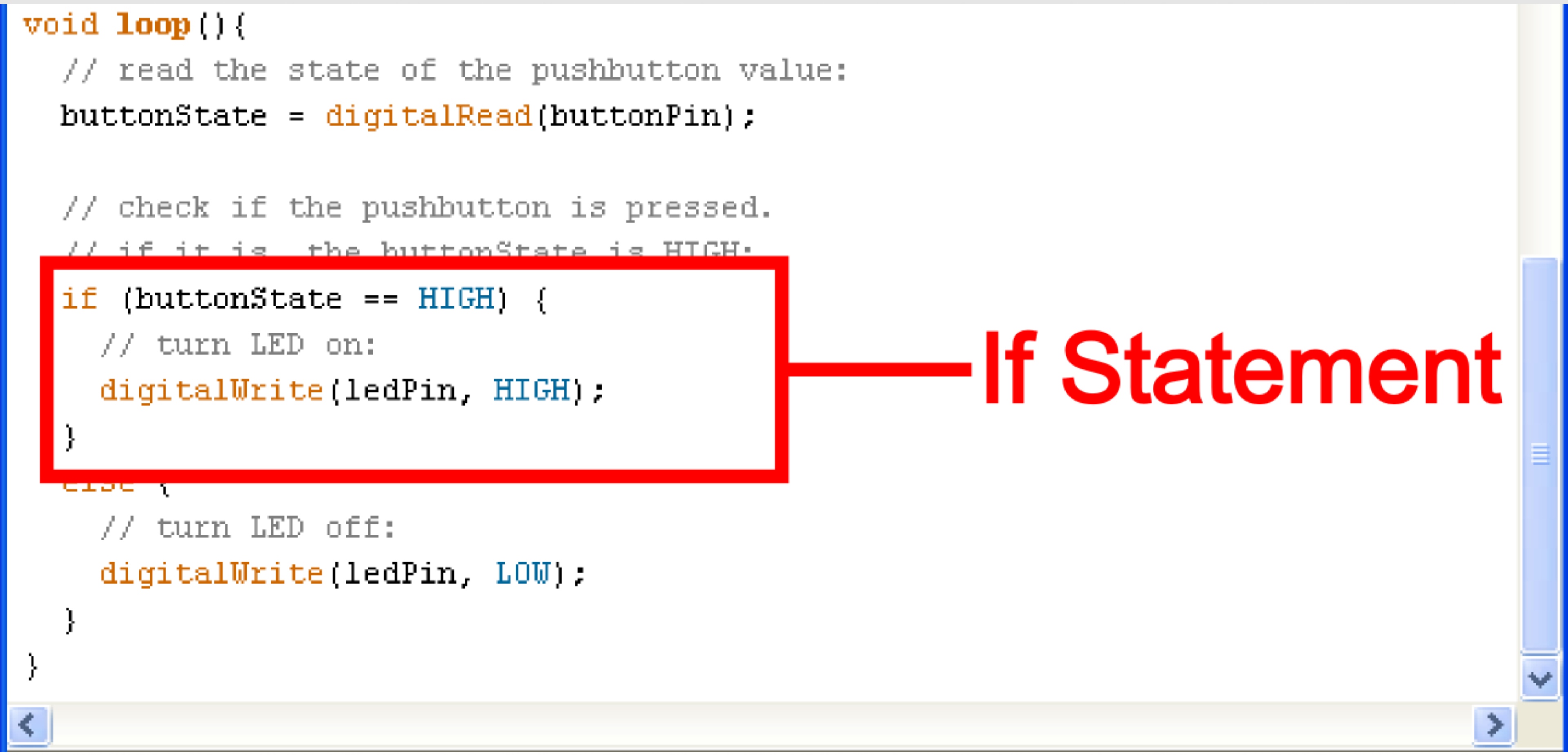
void setup ( ) {
attachInterrupt (interrupt, function, mode) }

- ***LOW*** whenever pin state is low
- ***CHANGE*** whenever pin changes value
- ***RISING*** whenever pin goes from low to high
- ***FALLING*** whenever pin goes from low to high

Don’t forget to CAPITALIZE

If Statements

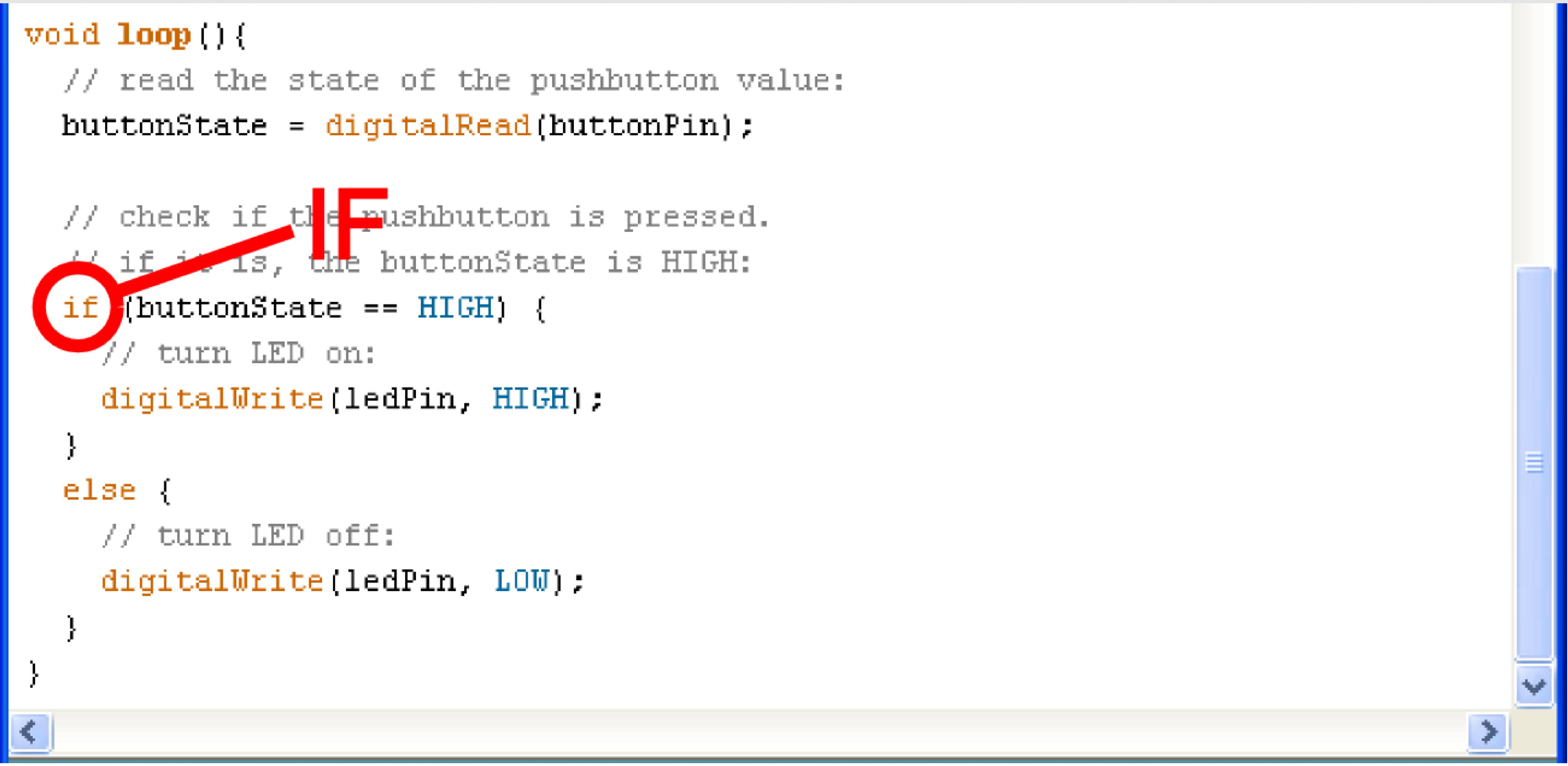
if (this is true) { do this; }



```
void loop(){
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed.
  // if it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  }
  else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

If

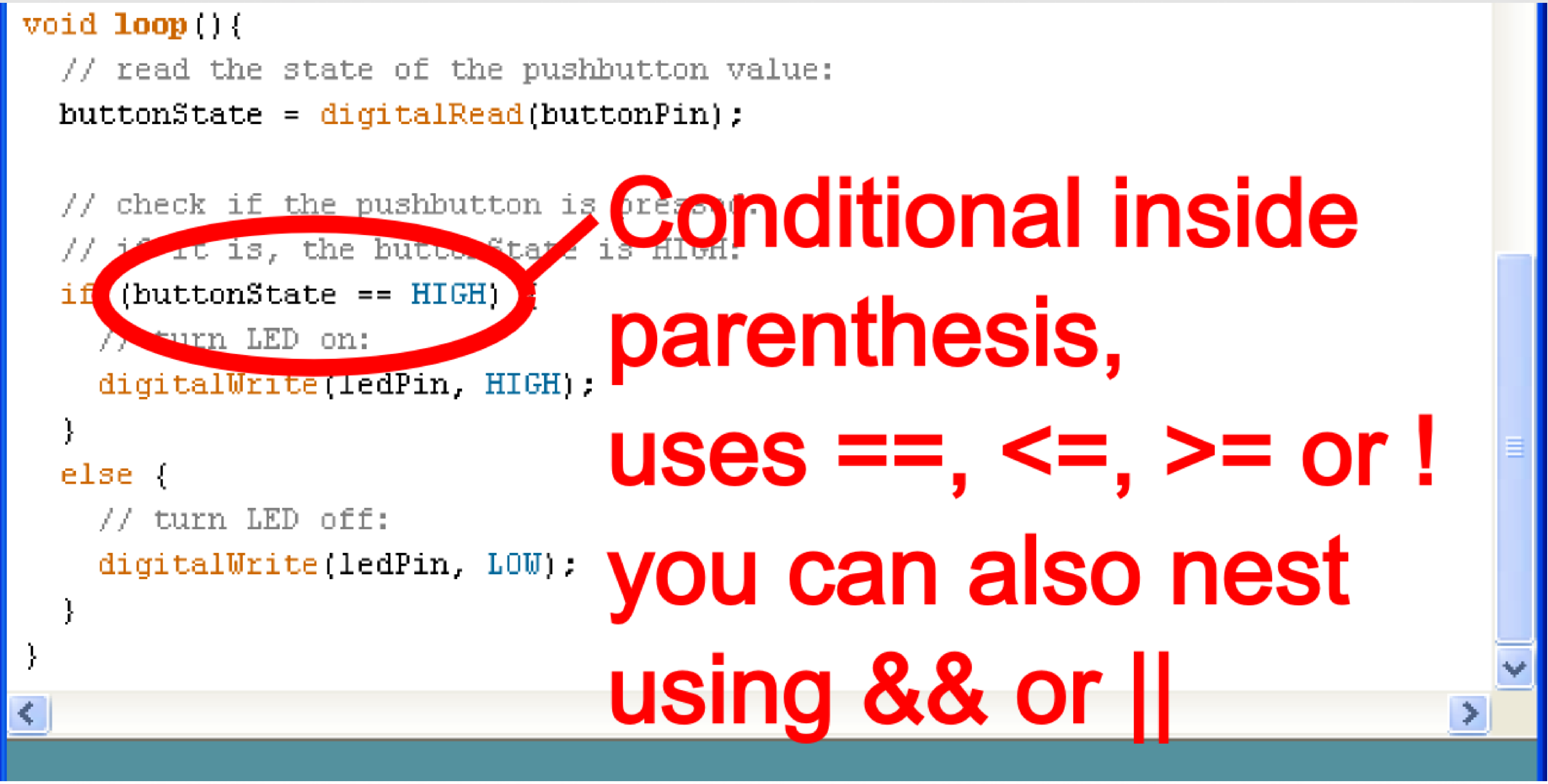
if (this is true) { do this; }



```
void loop(){
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed.
  // if it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  }
  else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

Conditional

if (this is true) { do this; }

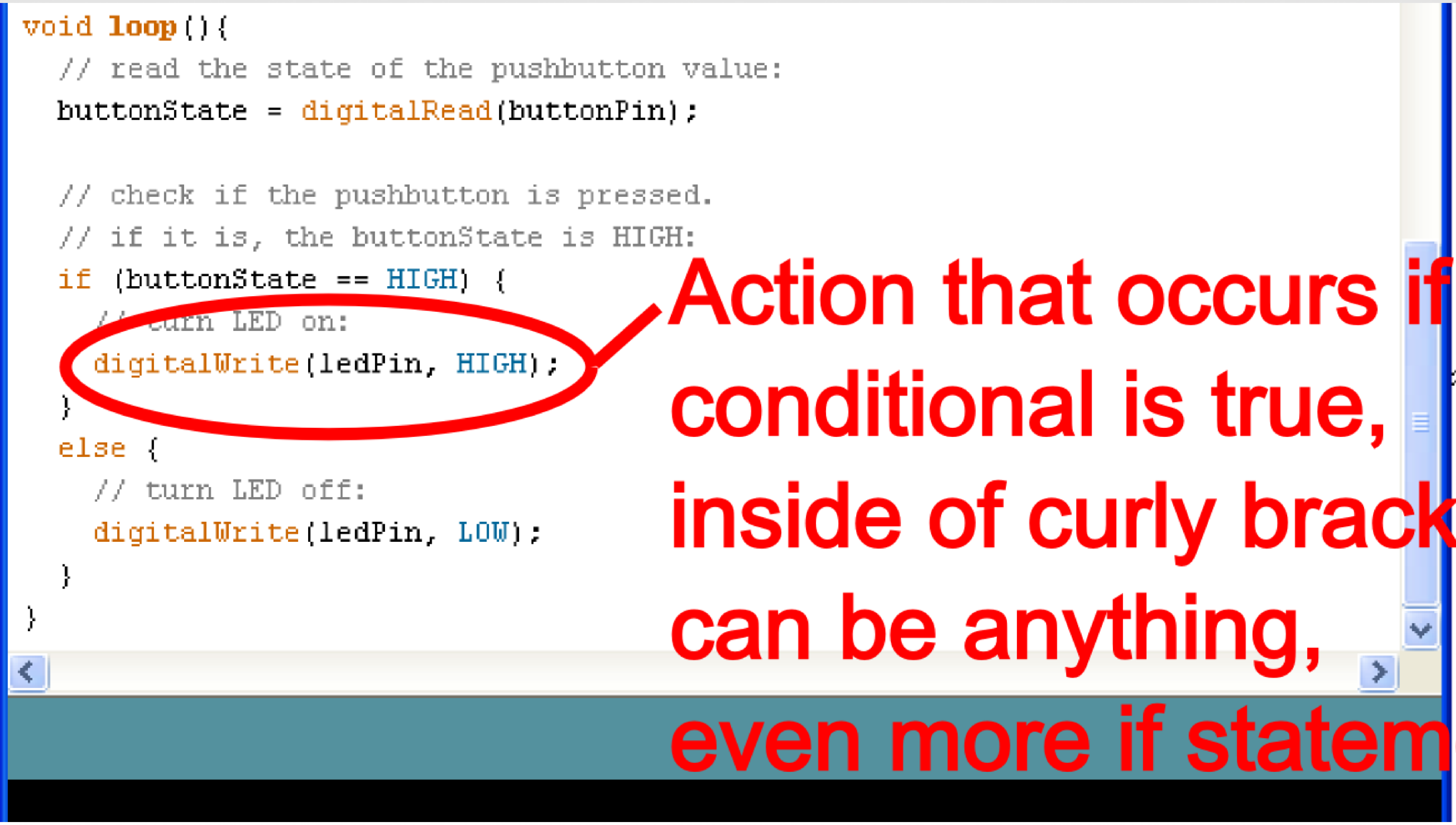


```
void loop(){
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed.
  // if it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  }
  else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

Conditional inside parenthesis, uses ==, <=, >= or ! you can also nest using && or ||

Action

if (this is true) { do this; }

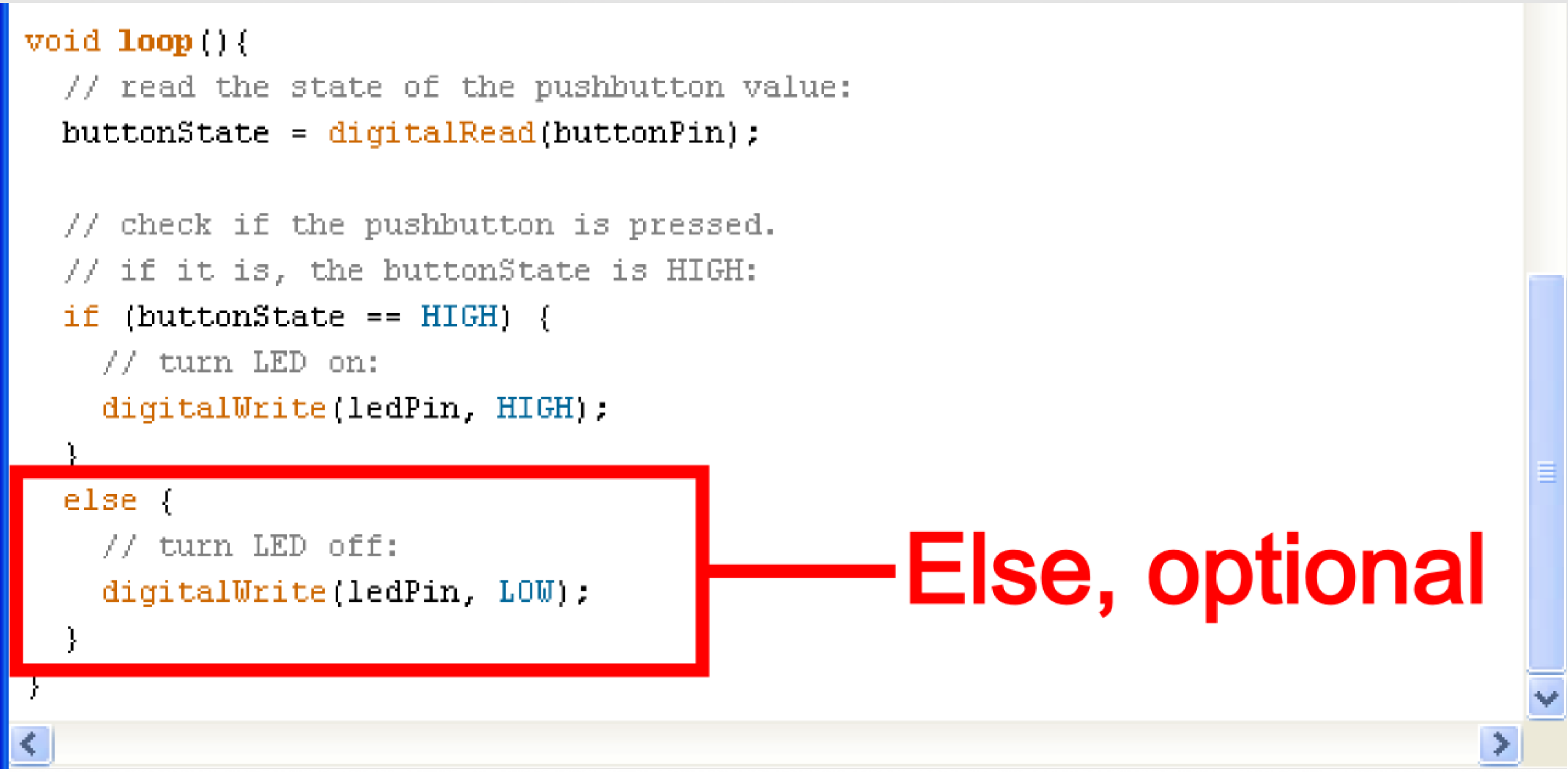


```
void loop(){
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed.
  // if it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  }
  else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

Action that occurs if conditional is true, inside of curly brackets, can be anything, even more if statements

Else

else { do this; }



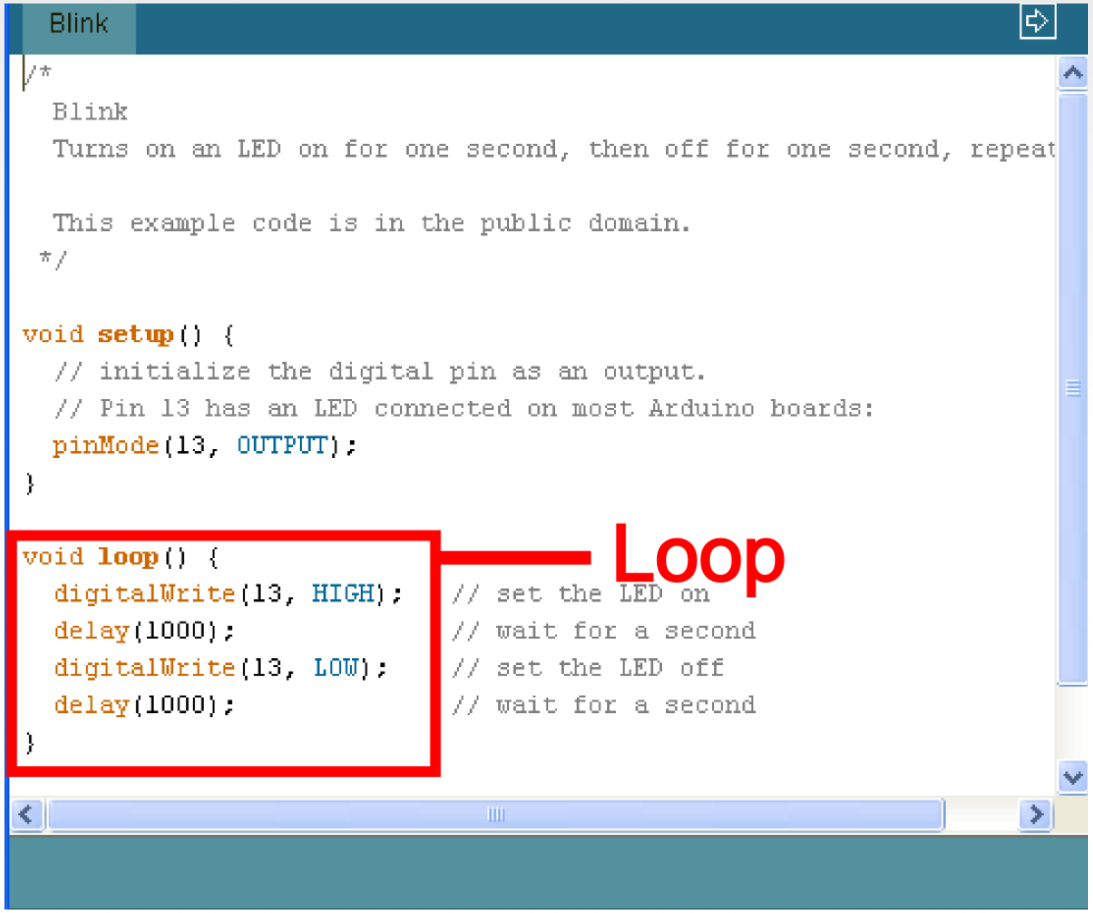
```
void loop(){
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed.
  // if it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  }
  else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

Basic Repetition

- loop
- For
- while

Basic Repetition

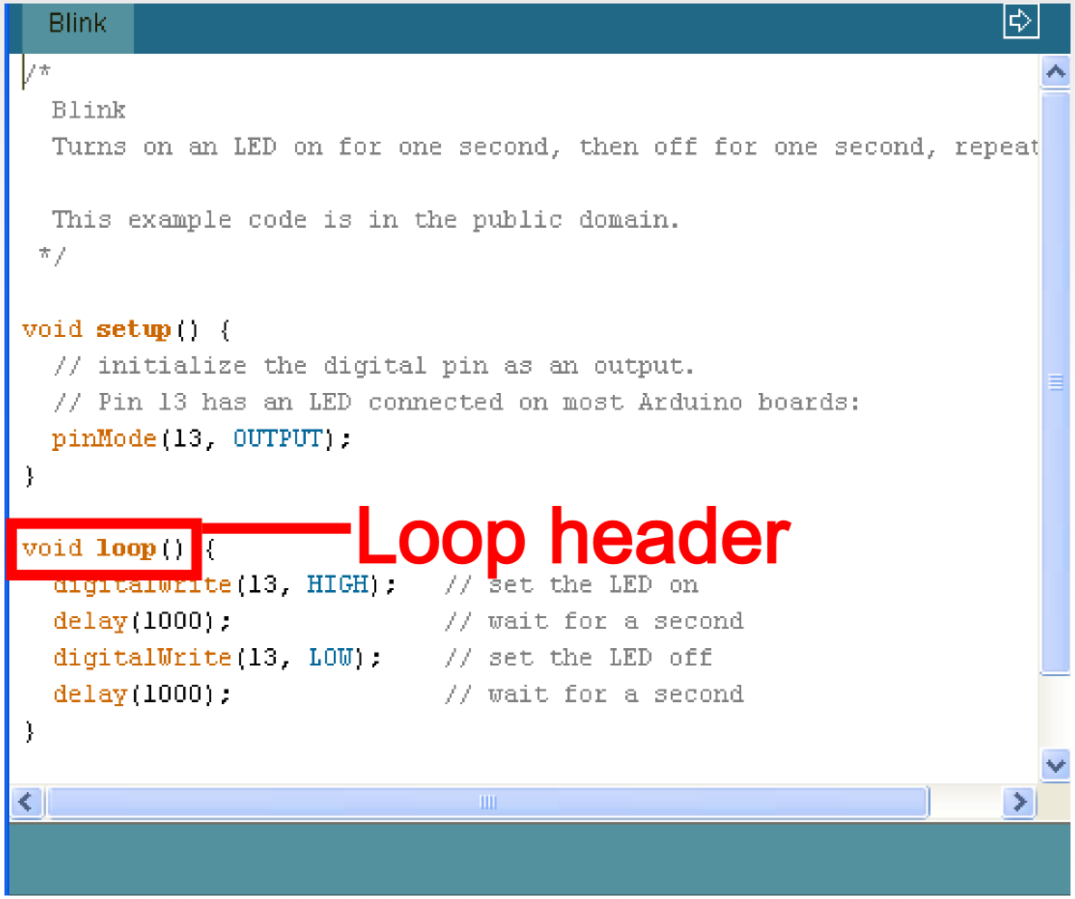
void loop ( ) { }



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

  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
}
```

Basic Repetition

void loop ( ) { }



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

  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
}
```

Basic Repetition

void loop ( ) { }

The “void” in the header is what the function will return (or spit out) when it happens, in this case it returns nothing so it is void

Basic Repetition

void loop ( ) { }

The “loop” in the header is what the function is called, sometimes you make the name up, sometimes (like loop) the function already has a name

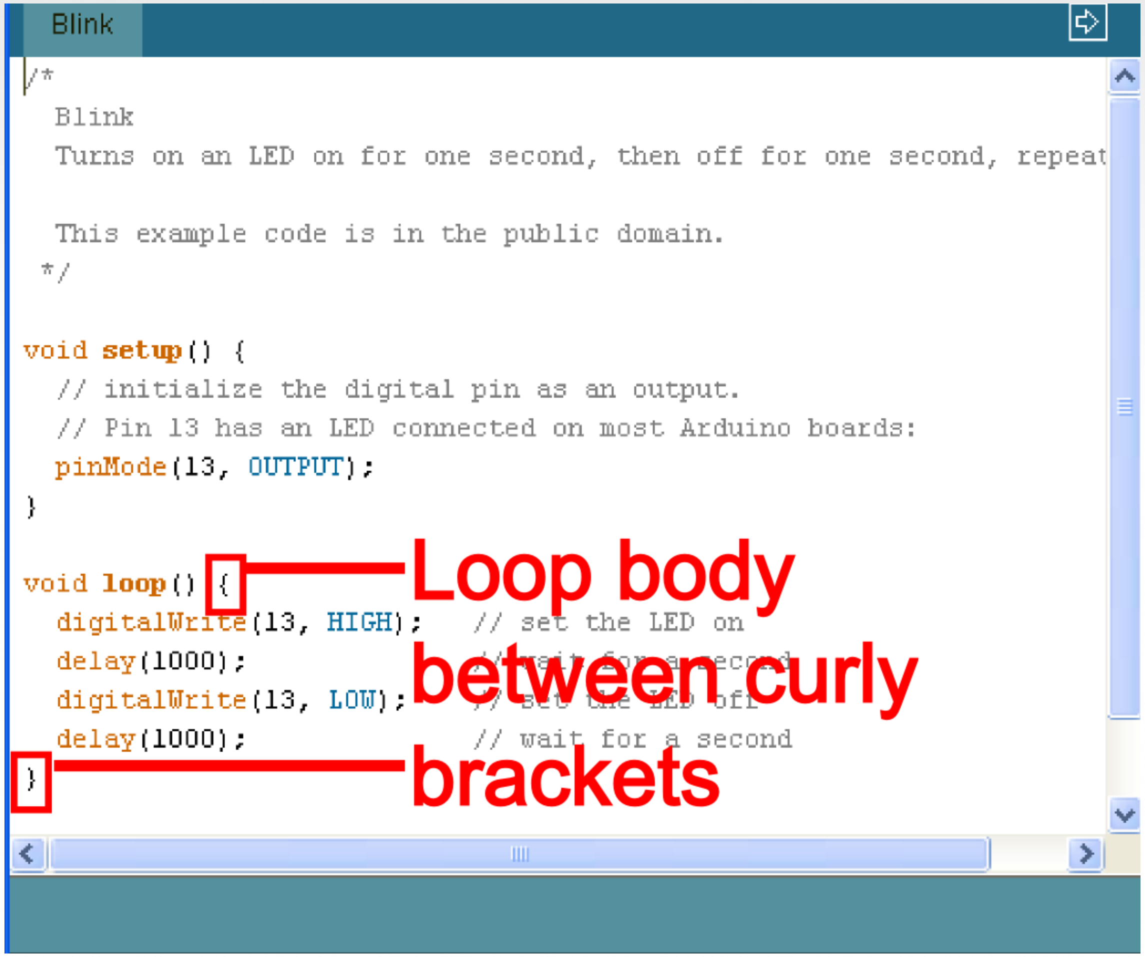
Basic Repetition

void loop ( ) { }

The “( )” in the header is where you declare any variables that you are “passing” (or sending) the function, the loop function is never “passed” any variables

Basic Repetition

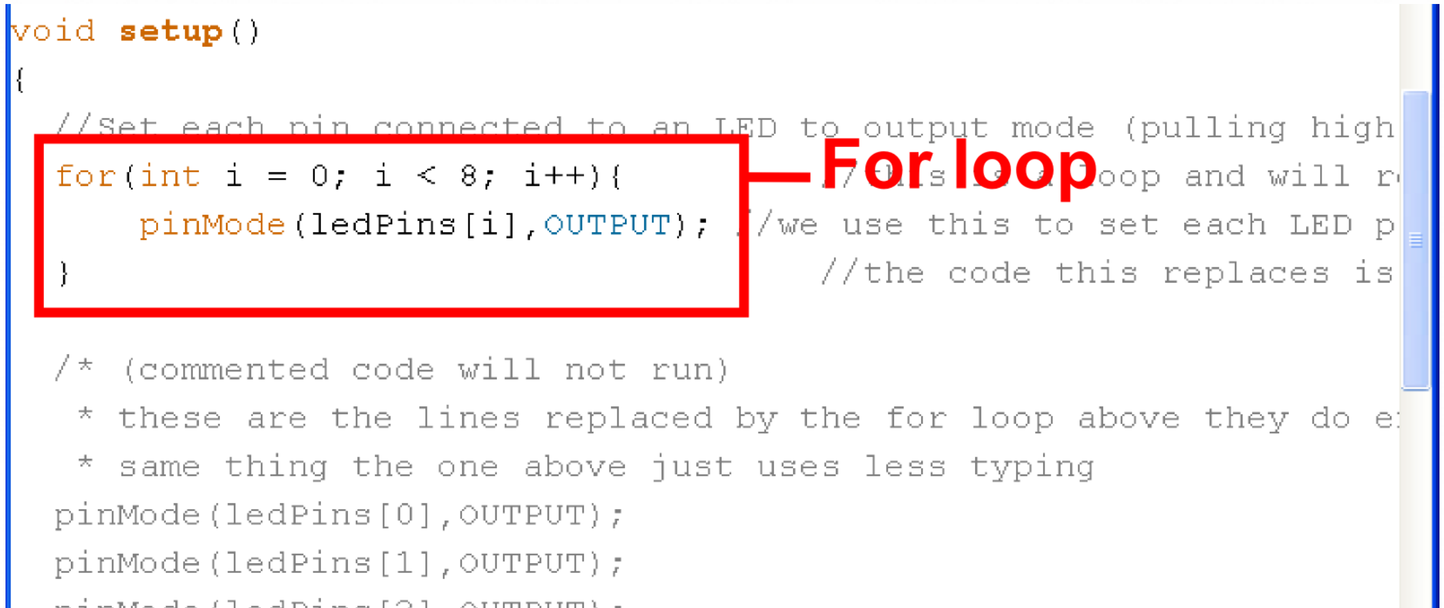
void loop ( ) { }



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

  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
}
```

Basic Repetition

for (int count = 0; count<10; count++)
{
//for action code goes here
//this could be anything
}



```
void setup()
{
  //Set each pin connected to an LED to output mode (pulling high
  for(int i = 0; i < 8; i++){         //this is a loop and will r
      pinMode(ledPins[i],OUTPUT);     //we use this to set each LED p
  }                                   //the code this replaces is

  /* (commented code will not run)
   * these are the lines replaced by the for loop above they do e
   * same thing the one above just uses less typing
  pinMode(ledPins[0],OUTPUT);
  pinMode(ledPins[1],OUTPUT);
```

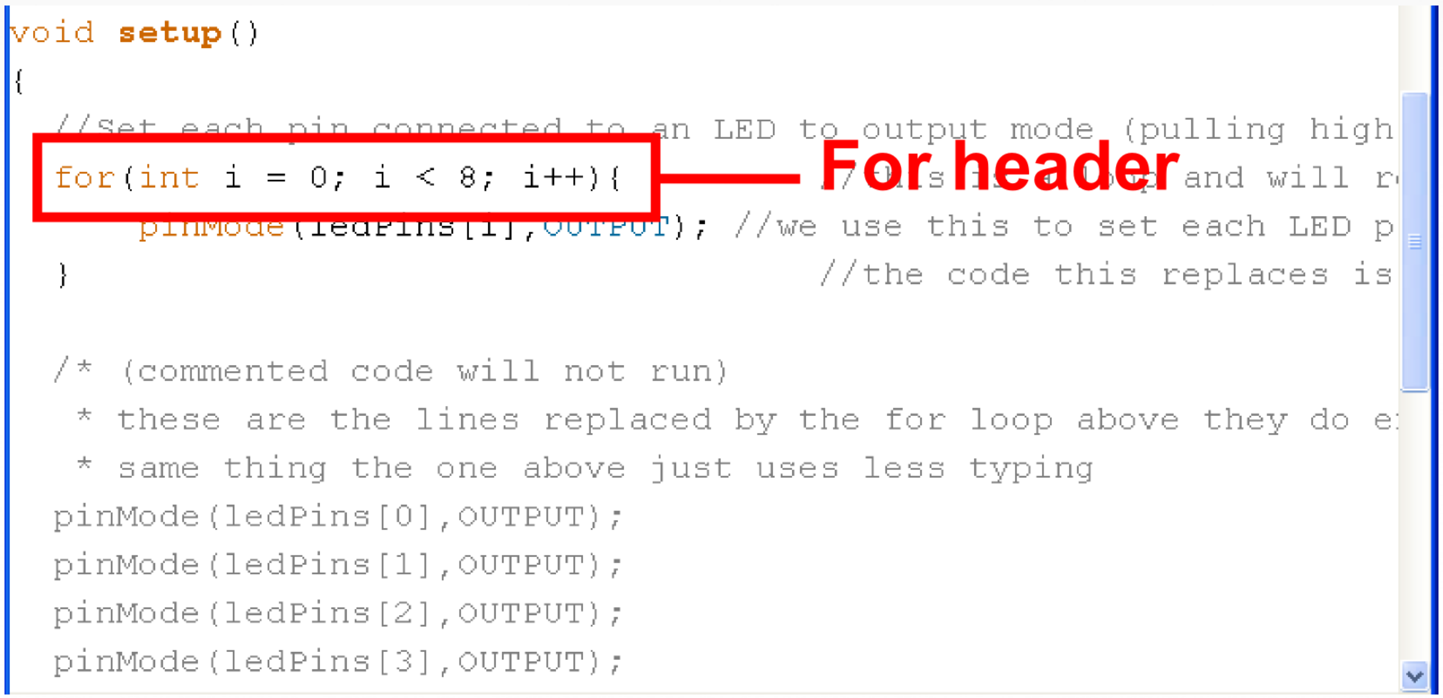
Basic Repetition

for (int count = 0; count<10; count++)

{

//for action code goes here

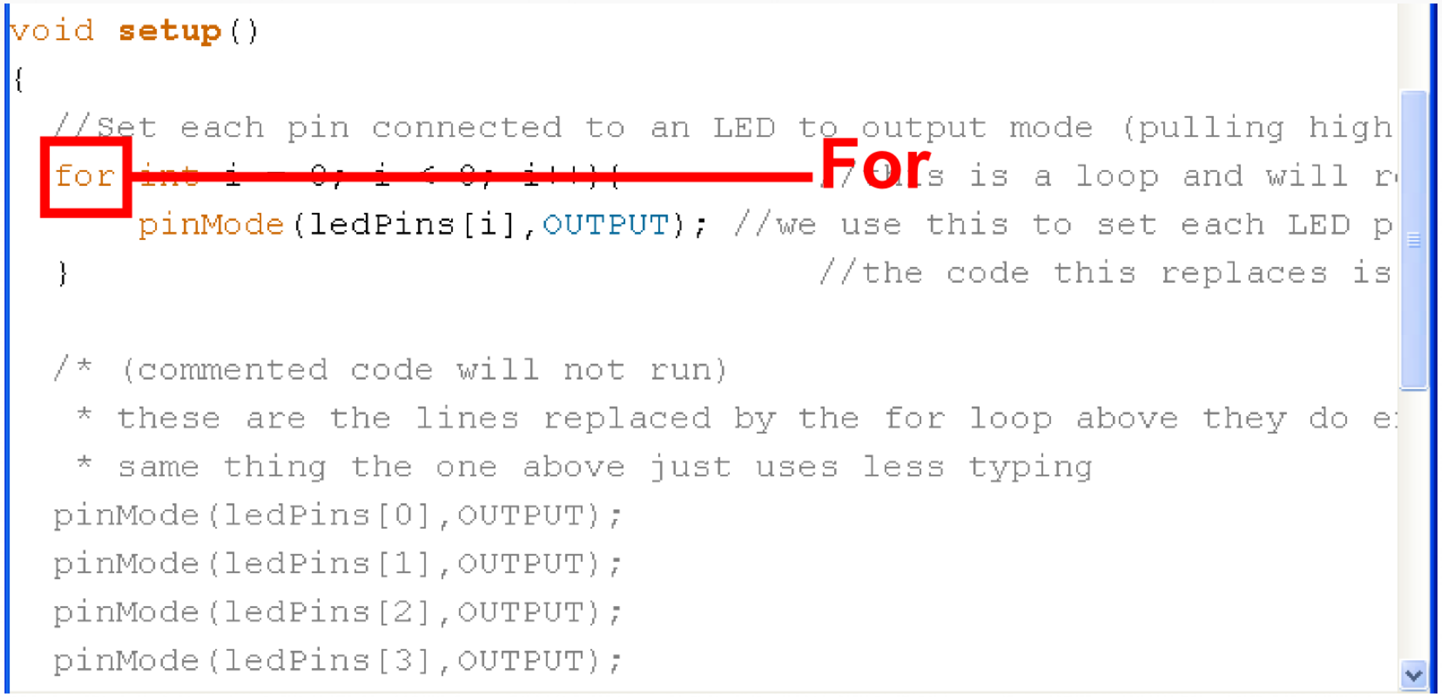
}



```
void setup()
{
  //Set each pin connected to an LED to output mode (pulling high
  for(int i = 0; i < 8; i++){          //this is a loop and will r
      pinMode(ledPins[i],OUTPUT); //we use this to set each LED p
  }                                    //the code this replaces is

  /* (commented code will not run)
   * these are the lines replaced by the for loop above they do e
   * same thing the one above just uses less typing
  pinMode(ledPins[0],OUTPUT);
  pinMode(ledPins[1],OUTPUT);
  pinMode(ledPins[2],OUTPUT);
  pinMode(ledPins[3],OUTPUT);
```

Basic Repetition

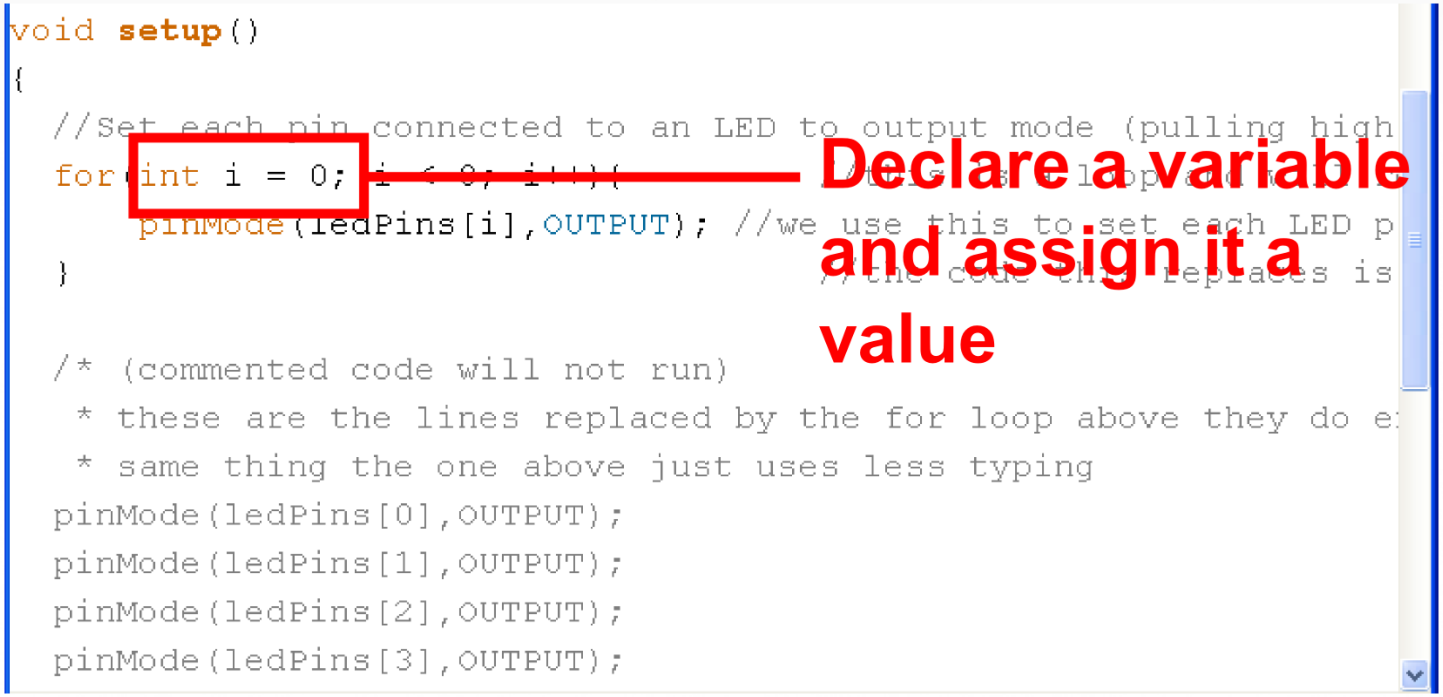
for (int count = 0; count<10; count++)
{
//for action code goes here
}



```
void setup()
{
  //Set each pin connected to an LED to output mode (pulling high
  for (int i = 0; i < 8; i++){      //this is a loop and will r
      pinMode(ledPins[i],OUTPUT); //we use this to set each LED p
  }                                   //the code this replaces is

  /* (commented code will not run)
   * these are the lines replaced by the for loop above they do e
   * same thing the one above just uses less typing
  pinMode(ledPins[0],OUTPUT);
  pinMode(ledPins[1],OUTPUT);
  pinMode(ledPins[2],OUTPUT);
  pinMode(ledPins[3],OUTPUT);
```

Basic Repetition

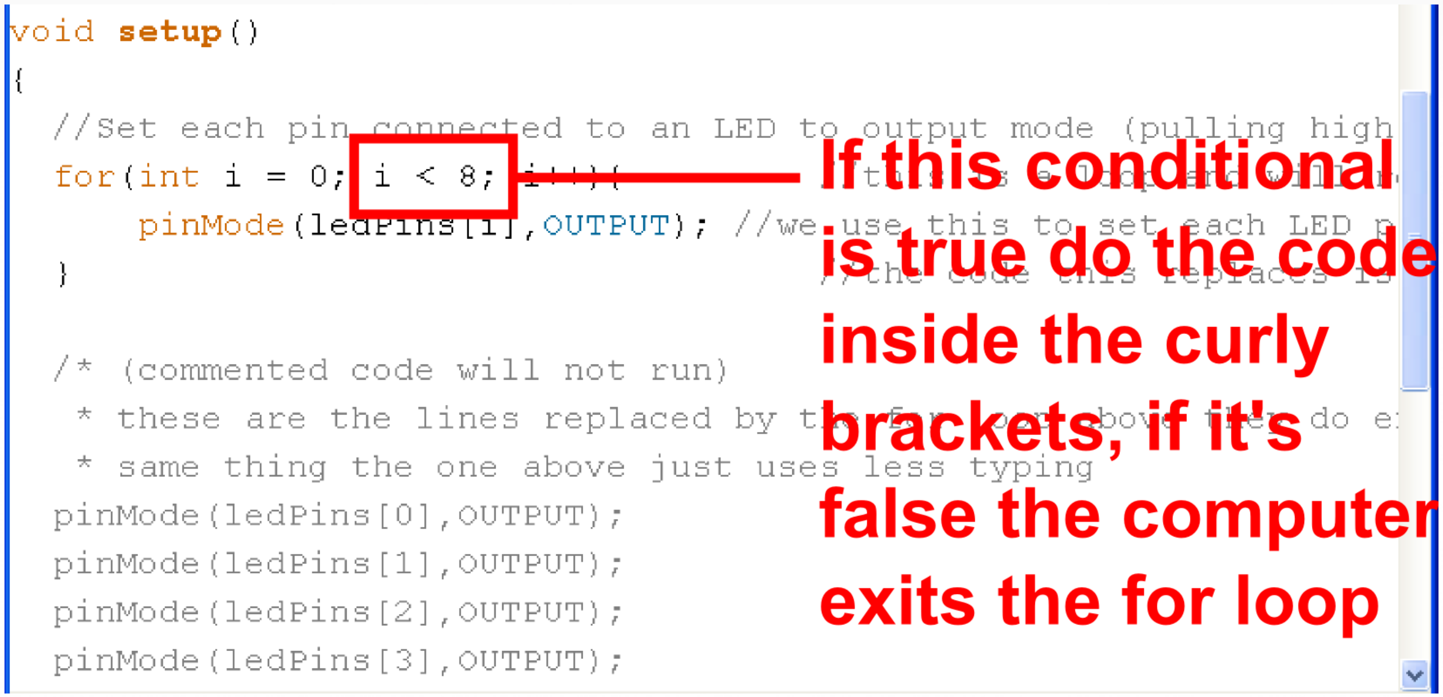
***for (int count = 0;* count<10; count++)**
{
//for action code goes here
}



```
void setup()
{
  //Set each pin connected to an LED to output mode (pulling high
  for(int i = 0; i < 8; i++){   //this is a loop and will
      pinMode(ledPins[i],OUTPUT); //we use this to set each LED p
  }                               //the code this replaces is

  /* (commented code will not run)
   * these are the lines replaced by the for loop above they do e
   * same thing the one above just uses less typing
  pinMode(ledPins[0],OUTPUT);
  pinMode(ledPins[1],OUTPUT);
  pinMode(ledPins[2],OUTPUT);
  pinMode(ledPins[3],OUTPUT);
```

Basic Repetition

for (int count = 0; count<10; count++)
{
//for action code goes here
}

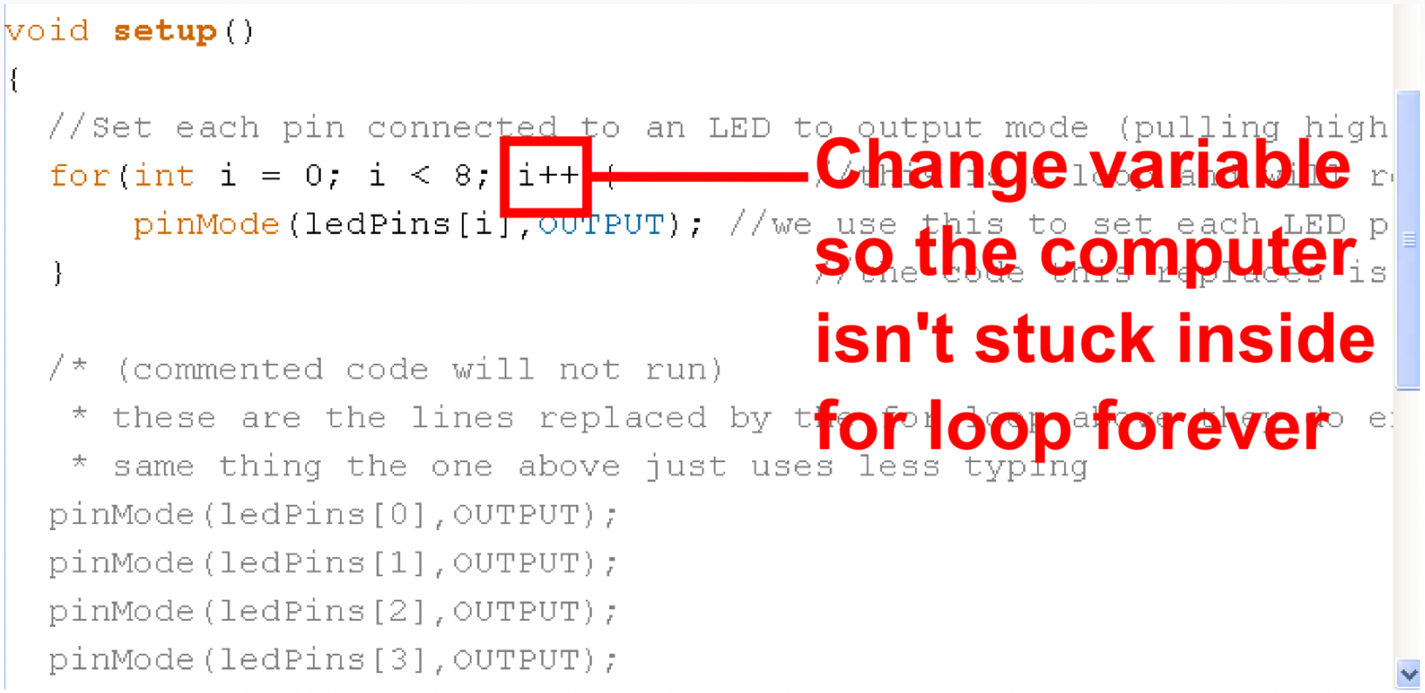


```
void setup()
{
  //Set each pin connected to an LED to output mode (pulling high
  for(int i = 0; i < 8; i++){ //this is a loop and will r
      pinMode(ledPins[i],OUTPUT); //we use this to set each LED p
  }                              //the code this replaces is

  /* (commented code will not run)
   * these are the lines replaced by the for loop above they do e
   * same thing the one above just uses less typing
  pinMode(ledPins[0],OUTPUT);
  pinMode(ledPins[1],OUTPUT);
  pinMode(ledPins[2],OUTPUT);
  pinMode(ledPins[3],OUTPUT);
```

If this conditional is true do the code inside the curly brackets, if it's false the computer exits the for loop

Basic Repetition

for (int count = 0; count<10; count++)
{
//for action code goes here
}



```
void setup()
{
  //Set each pin connected to an LED to output mode (pulling high
  for(int i = 0; i < 8; i++){
      pinMode(ledPins[i],OUTPUT); //we use this to set each LED p
  }

  /* (commented code will not run)
   * these are the lines replaced by the for loop above they do e
   * same thing the one above just uses less typing
  pinMode(ledPins[0],OUTPUT);
  pinMode(ledPins[1],OUTPUT);
  pinMode(ledPins[2],OUTPUT);
  pinMode(ledPins[3],OUTPUT);
```

Change variable so the computer isn't stuck inside for loop forever

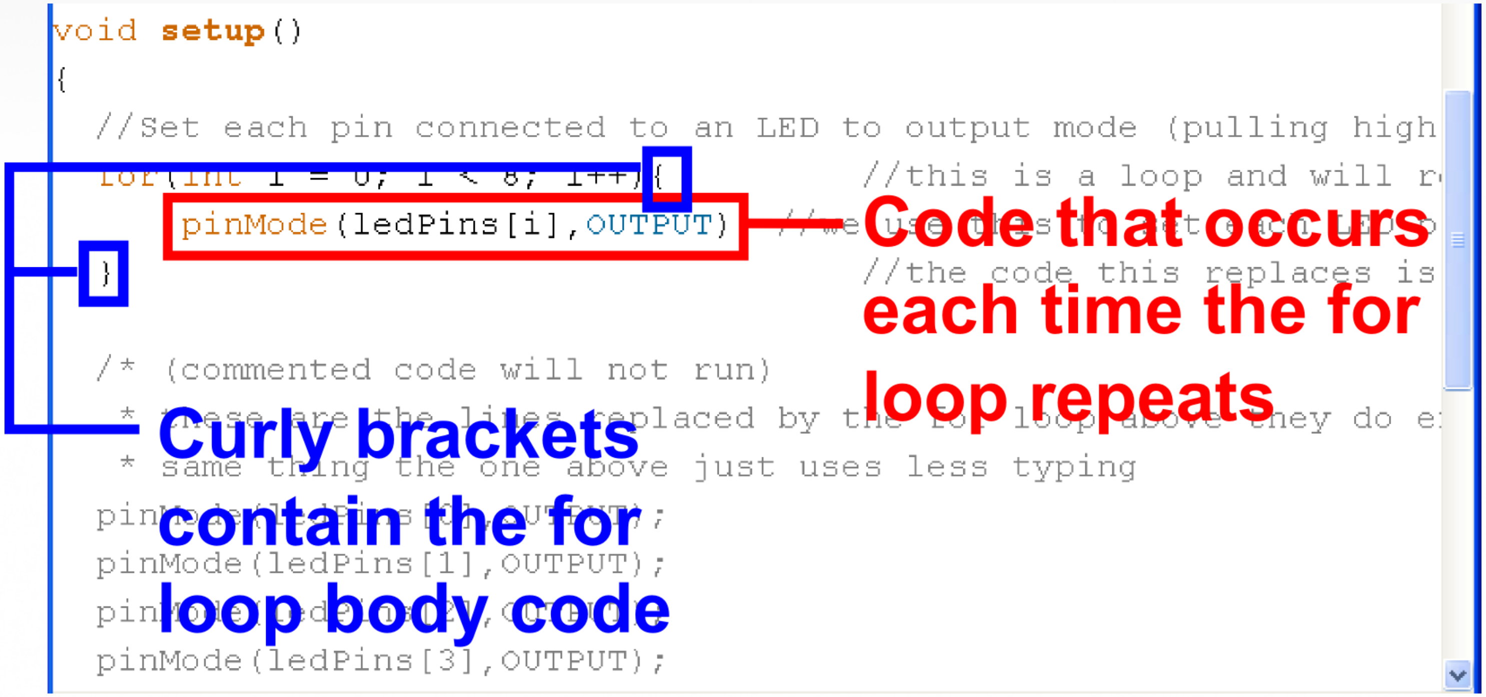
Basic Repetition

for (int count = 0; count<10; count++)

{

//for action code goes here

}


Code that occurs each time the for loop repeats
Curly brackets contain the for loop body code

```
void setup()
{
  //Set each pin connected to an LED to output mode (pulling high
  for(int i = 0; i < 8; i++){          //this is a loop and will r
      pinMode(ledPins[i],OUTPUT)      //we use this to set each LED to o
  }                                    //the code this replaces is

  /* (commented code will not run)
   * these are the lines replaced by the for loop above they do e
   * same thing the one above just uses less typing
  pinMode(ledPins[0],OUTPUT);
  pinMode(ledPins[1],OUTPUT);
  pinMode(ledPins[2],OUTPUT);
  pinMode(ledPins[3],OUTPUT);
```

Basic Repetition

```
while ( count<10 )
{
//while action code goes here
}
```

Basic Repetition

```
while ( count<10 )
{
//while action code goes here
//should include a way to change count
//variable so the computer is not stuck
//inside the while loop forever
}
```

Basic Repetition

```
while ( count<10 )
{
//looks basically like a “for” loop
//except the variable is declared before
//and incremented inside the while
//loop
}
```

Basic Repetition
Or maybe:

```
while ( digitalRead(buttonPin)==1 )
{
//instead of changing a variable
//you just read a pin so the computer
//exits when you press a button
//or a sensor is tripped
}
```

Questions?

sparkfun™ ELECTRONICS

www.sparkfun.com
6175 Longbow Drive, Suite 200
Boulder, Colorado 80301