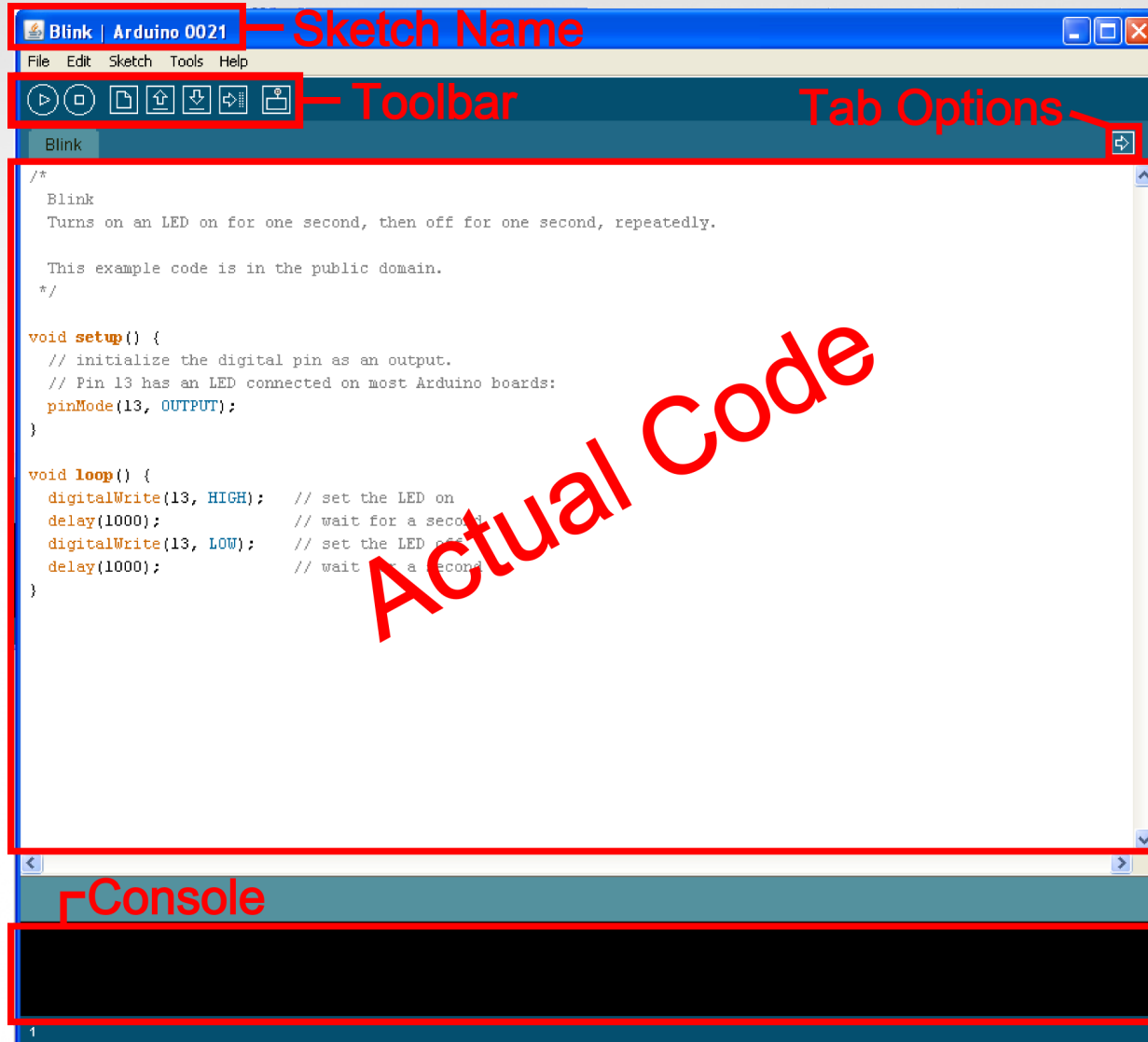




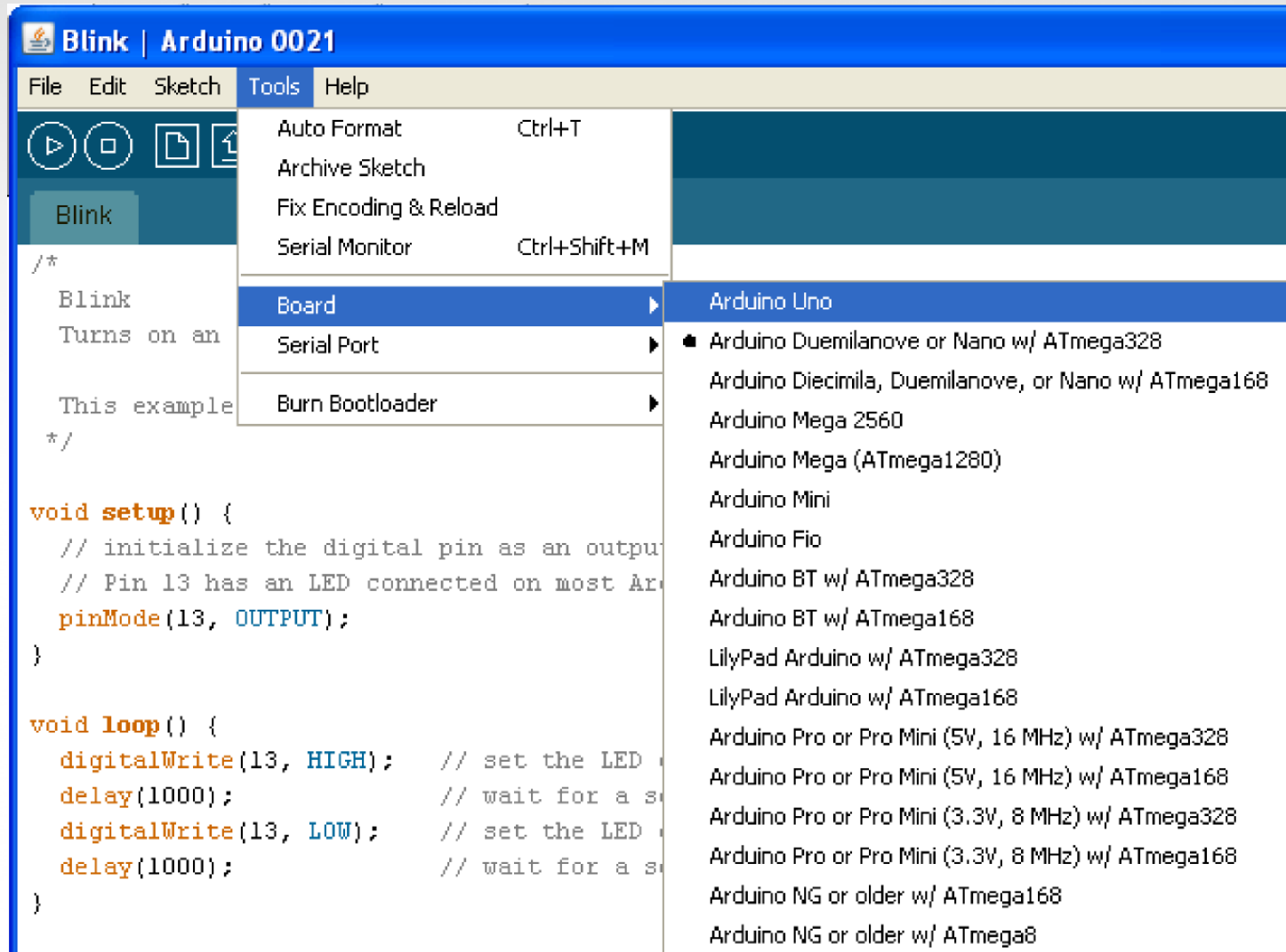
Code

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

The Arduino Environment



Board Type



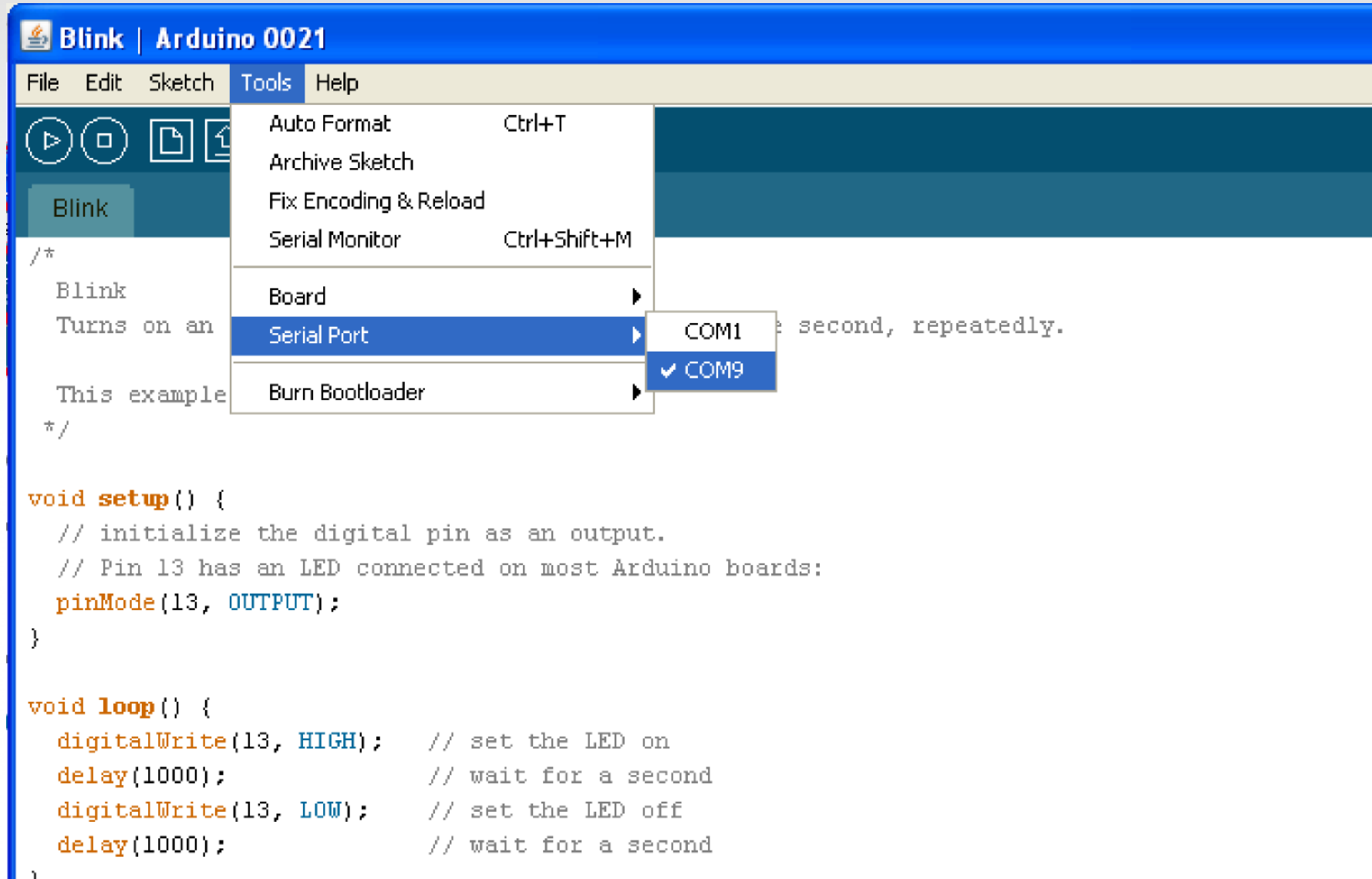
The screenshot shows the Arduino IDE interface. The title bar reads "Blink | Arduino 0021". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". The "Tools" menu is open, showing options: "Auto Format" (Ctrl+T), "Archive Sketch", "Fix Encoding & Reload", "Serial Monitor" (Ctrl+Shift+M), "Board", "Serial Port", and "Burn Bootloader". The "Board" option is selected, opening a submenu with the following board types:

- 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

The code editor shows the following code:

```
/*  
  Blink  
  Turns on an LED on for one second, then off for one second, repeating.  
  This example is adapted from the Arduino tutorial.  
  */  
  
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  
}
```

Serial Port / COM Port



The screenshot shows the Arduino IDE interface for a sketch named "Blink" on an "Arduino 0021" board. The "Tools" menu is open, and the "Serial Port" option is selected, which has opened a submenu where "COM9" is highlighted with a checkmark. The code editor shows the standard Arduino Blink sketch code.

```
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Board
Serial Port
Burn Bootloader
COM1
COM9

/*
 * 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
}
```

The Environment

The image shows the Arduino IDE interface with several components highlighted by red boxes and labels:

- Sketch Name:** A red box highlights the title bar text "Blink | Arduino 0021".
- Toolbar:** A red box highlights the menu bar (File, Edit, Sketch, Tools, Help) and the toolbar below it.
- Upload:** A red arrow points from the upload icon (a right-pointing triangle) in the toolbar to the text "Upload".
- Save:** A red arrow points from the save icon (a floppy disk) in the toolbar to the text "Save".
- Open:** A red arrow points from the open icon (a folder) in the toolbar to the text "Open".
- New:** A red arrow points from the new icon (a document with a plus sign) in the toolbar to the text "New".
- Stop:** A red arrow points from the stop icon (a square) in the toolbar to the text "Stop".
- Compile:** A red arrow points from the compile icon (a document with a lightning bolt) in the toolbar to the text "Compile".
- Serial Monitor:** A red arrow points from the serial monitor icon (a computer monitor) in the toolbar to the text "Serial Monitor".

The main code editor area contains the following code:

```
/*
 * 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
}
```

Actual Code

Parts of the Sketch

The image shows a screenshot of the Arduino IDE interface. The title bar reads "Blink | Arduino 0021". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for running, stopping, saving, and uploading. The main text area contains the following code:

```
/*  
  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$
/*
  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 high to low

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 Statement

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);
  }
}
```

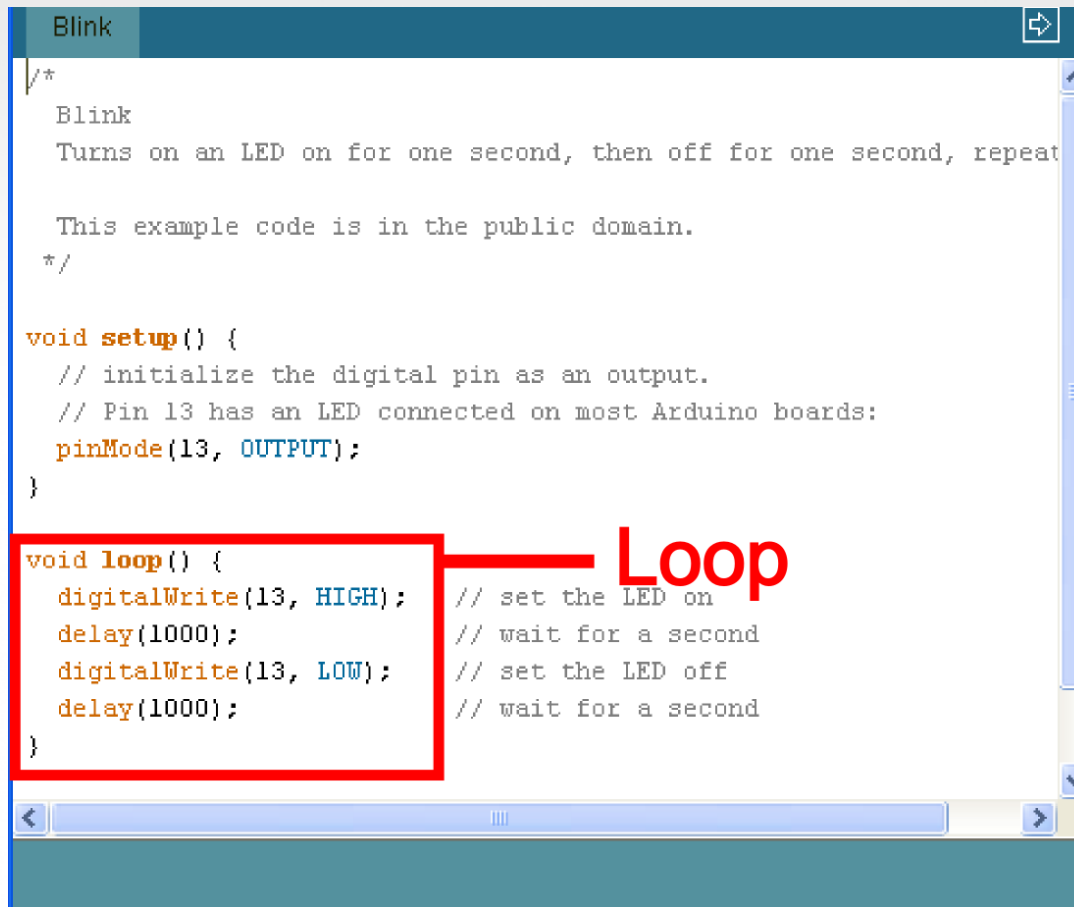
Else, optional

Basic Repetition

- loop
- For
- while

Basic Repetition

void loop () {}



```
Blink
/*
  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
/*
  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
}
```

Loop header

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
/*
  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
}
```

Loop body
between curly
brackets

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++){ // For loop  
    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);
```

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++){ For header //pins 0-7 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 < 9; i++) { For 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 < 10; i++){  
    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);
```

**Declare a variable
and assign it a
value**

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);
```

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 after they've 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  
}
```

```
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 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);
```

Code that occurs each time the for loop repeats

Curly brackets contain the for loop body code

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?



www.sparkfun.com

6175 Longbow Drive, Suite 200

Boulder, Colorado 80301