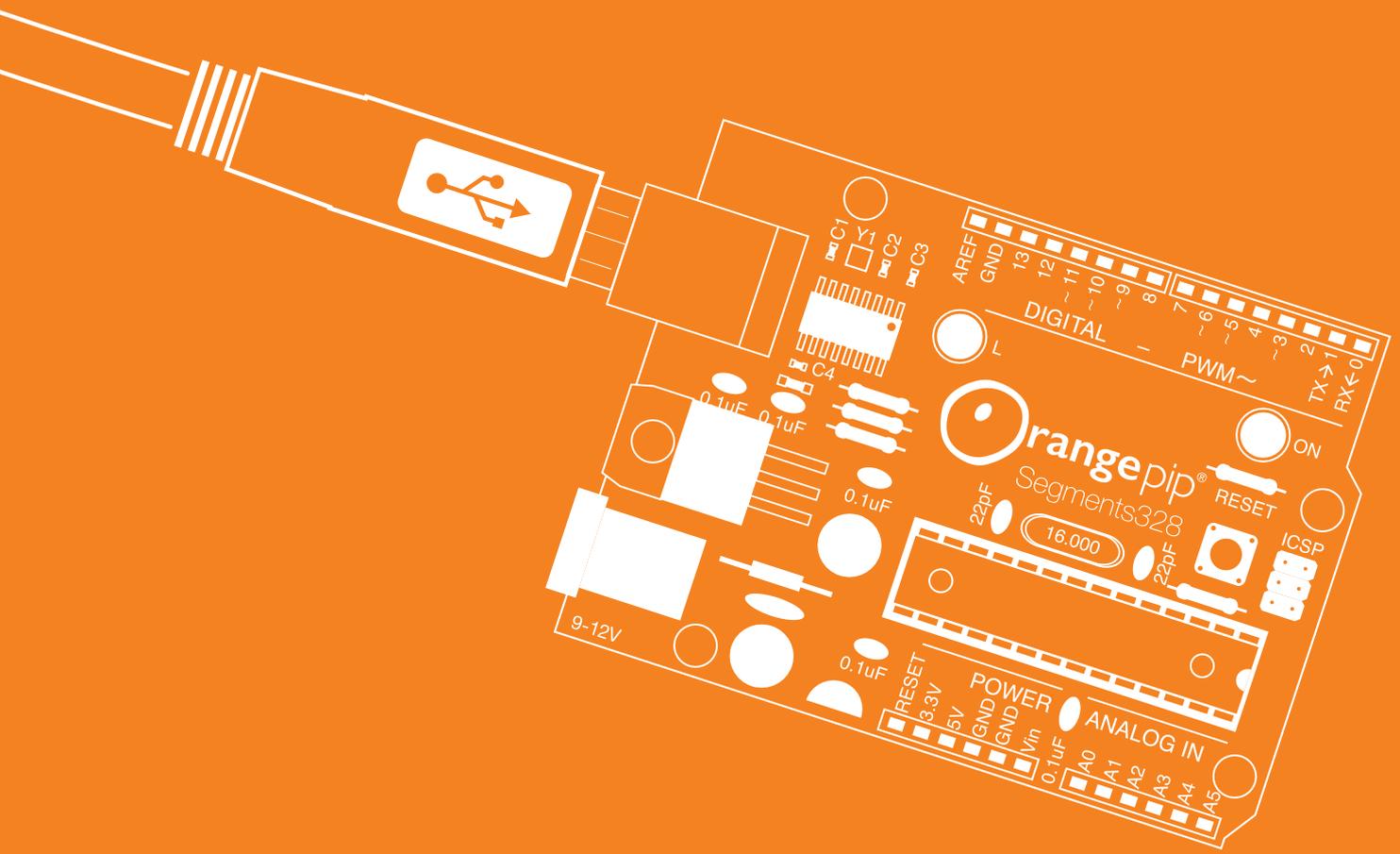


How to program Orangepip Segments328 with your PC



How to program Orangepip Segments328 with your PC

You've assembled your Orangepip Segments328 board. Now you need to connect it to your PC or laptop and start programming it. The great thing about the Orangepip Segments328 is that it is an Arduino compatible board that can be programmed with the Arduino IDE.

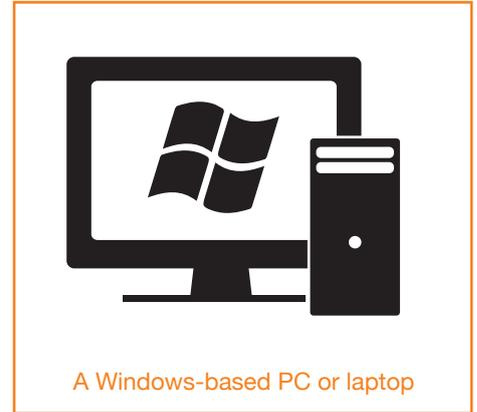
Equipment you will need ...



Orangepip Segments328



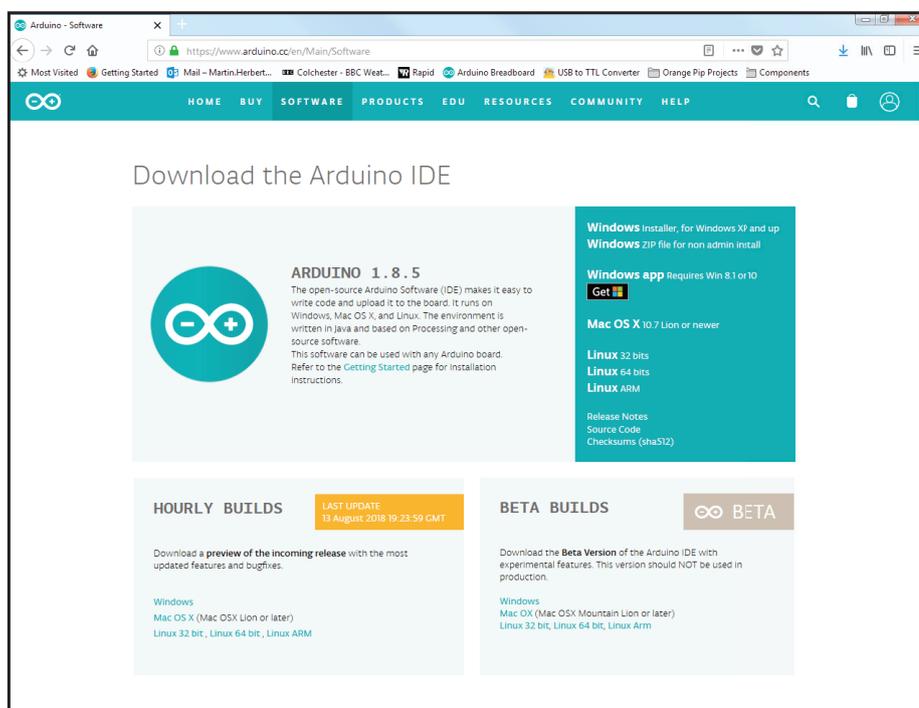
USB A to B cable



A Windows-based PC or laptop

1.0 Downloading and installing the Arduino IDE

Follow these simple instructions to get your Orangepip Segments328 up and running. Navigate to <https://www.arduino.cc/en/Main/Software>



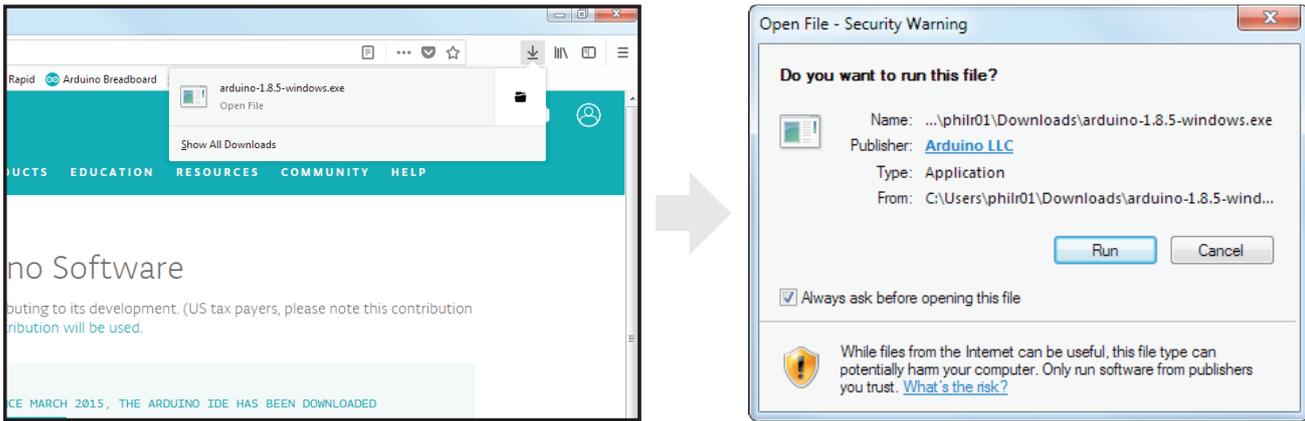
Click on “Windows Installer, for Windows XP and up”.

Arduino will ask for a contribution , but you can download for free. Click “Just Download”.

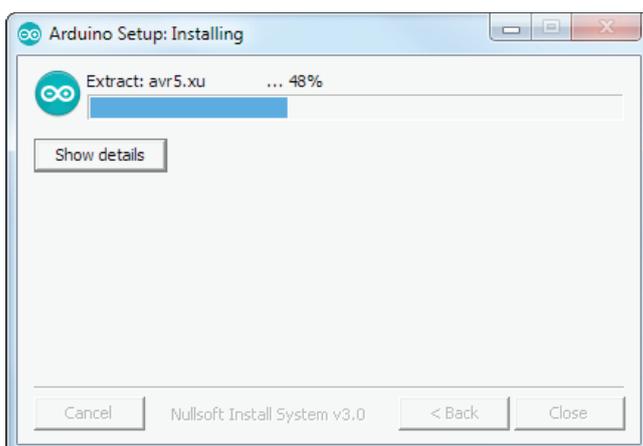
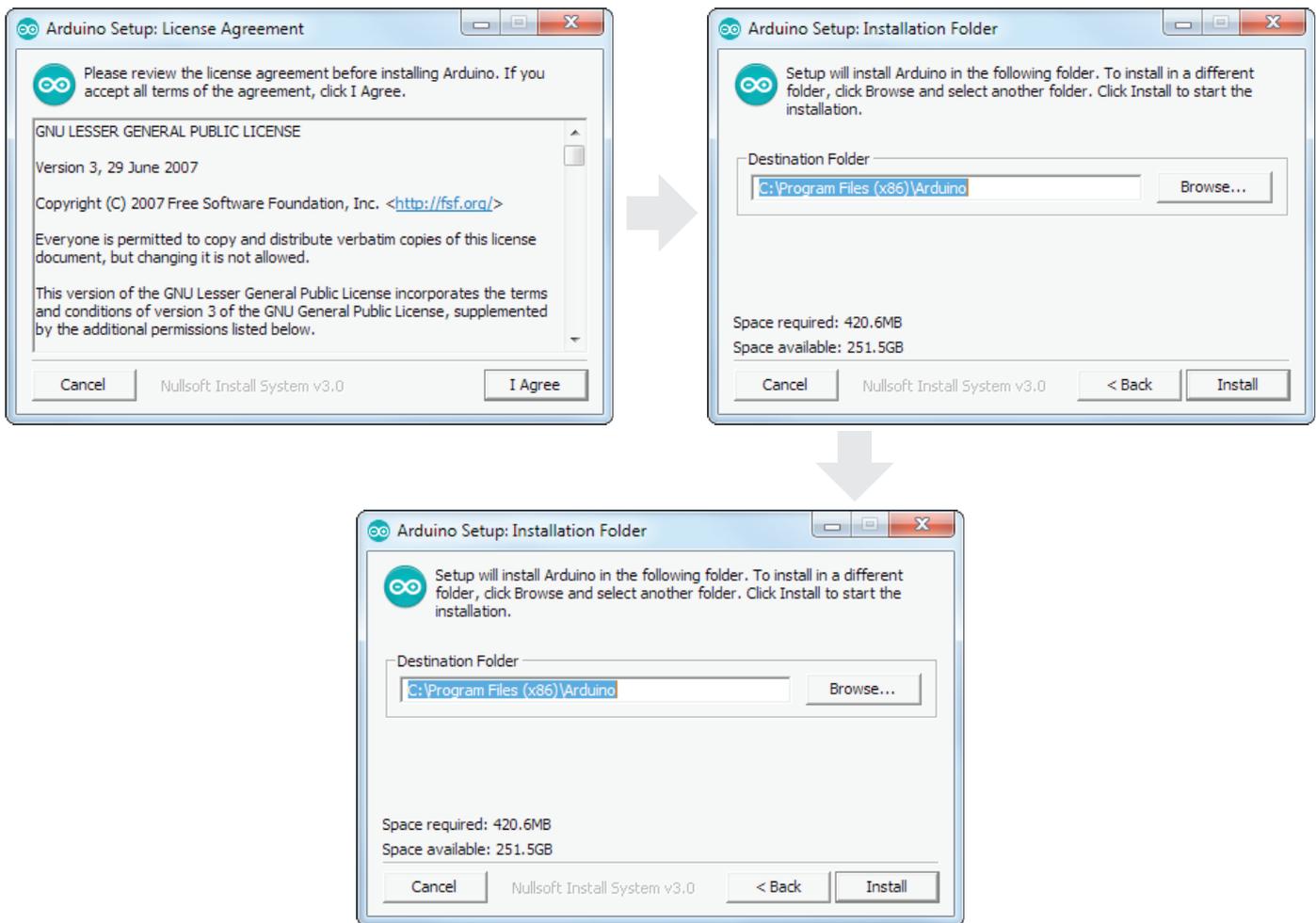
Your browser will ask where to save the file, click “Save File” and select a suitable location.



Once the download has completed you can access it via your browser, or navigate to where you saved it. Click “Run”.



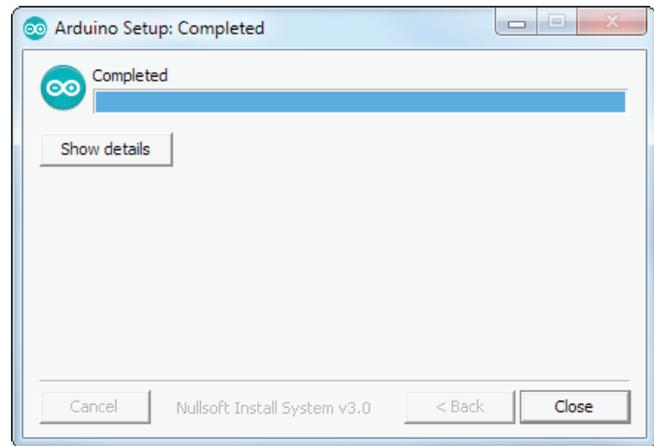
Follow the install instructions, clicking “I Agree”, “Next”, and “Install”.



The installer will start unpacking.



After a while, the installer will complete.
You are now ready to open the Arduino IDE.



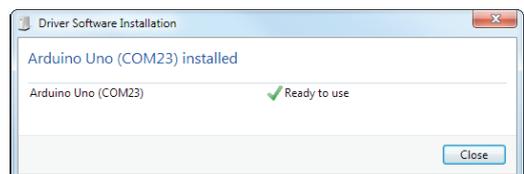
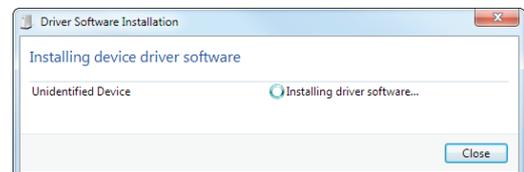
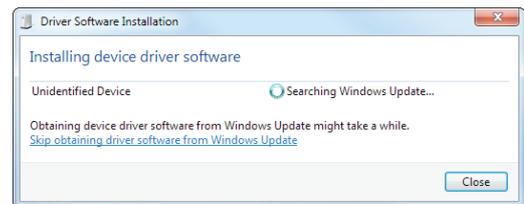
2.0 Start a Sketch

Double click on the Arduino icon on the desktop.



Connect the Orangepip Segments328 board to your PC or laptop using the USB cable.

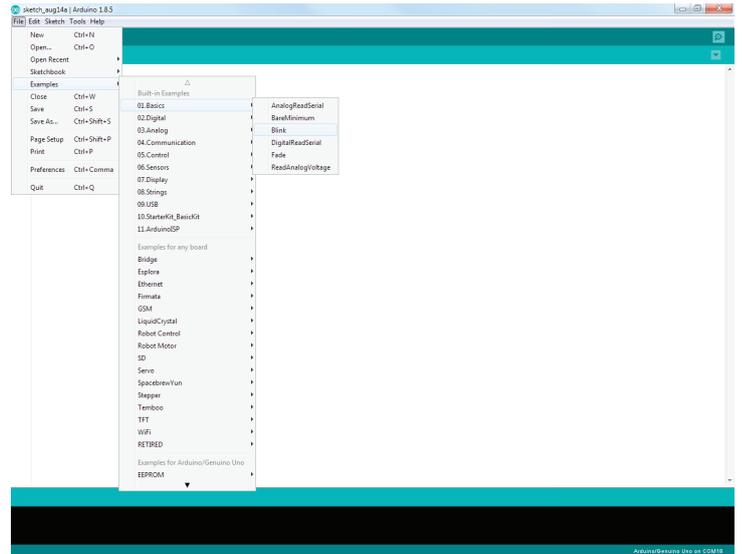
- The power LED on the right side of the board will light to indicate that the board now has power.
- Windows will begin to automatically install the drivers and assign the Orangepip board a COM number.
- If for any reason the drivers fail to install follow the manual process at the end of the document.



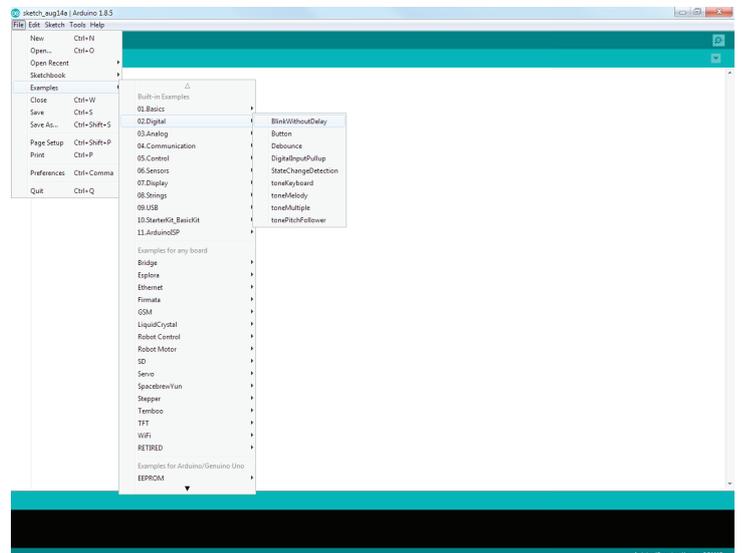
3.0 Uploading Blink

In true traditional Arduino fashion, upload the Blink sketch.

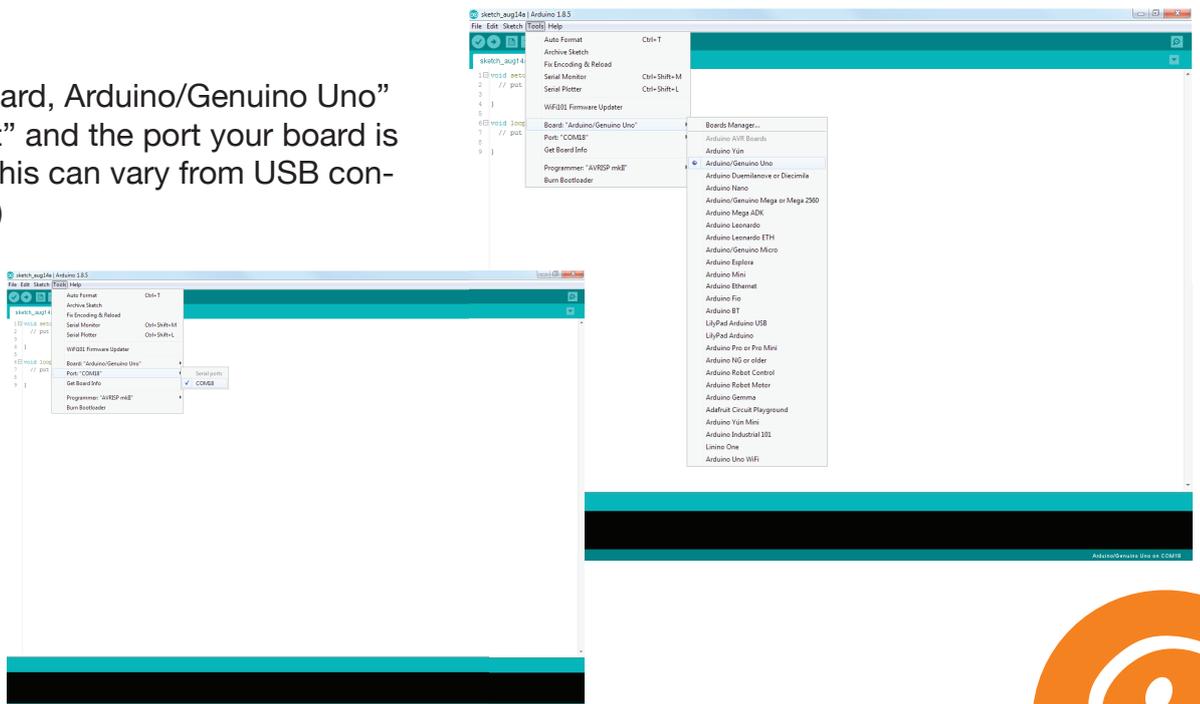
This can be located under
“File, Examples, 01. Basics, Blink”.



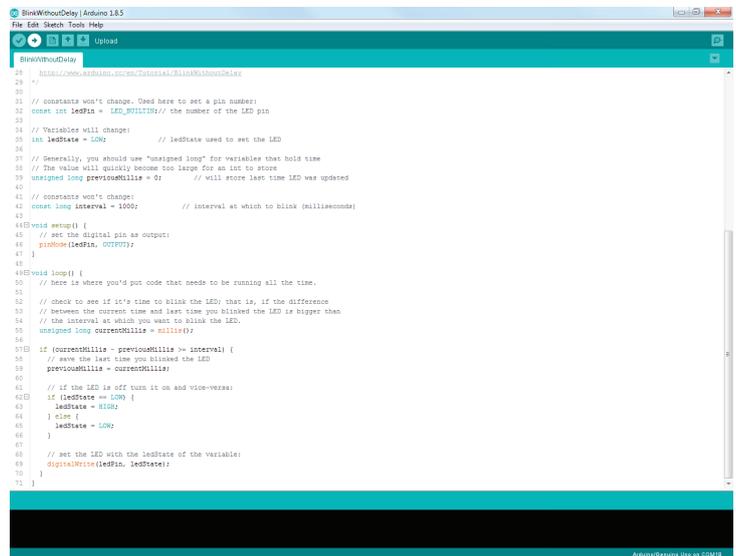
An alternative Blink can be located
under “File, Examples, 02. Digital,
BlinkWithoutDelay”.



Click “Tools, Board, Arduino/Genuino Uno”
and “Tools, Port” and the port your board
is connected to (This can vary from USB
connection and PC)



Click "Upload"



```
28 //  
29 // constants won't change. Used here to set a pin number:  
30 const int ledPin = LED_BUILTIN; // the number of the LED pin  
31  
32 // Variables will change:  
33 int ledState = LOW; // ledState used to set the LED  
34  
35 // Generally, you should use "unsigned long" for variables that hold time  
36 // The value will quickly become too large for an int to store  
37 unsigned long previousMillis = 0; // will store last time LED was updated  
38  
39 // constants won't change:  
40 const long interval = 1000; // interval at which to blink (milliseconds)  
41  
42 void setup() {  
43   // set the digital pin as output:  
44   pinMode(ledPin, OUTPUT);  
45 }  
46  
47 void loop() {  
48   // here is where you'd put code that needs to be running all the time.  
49  
50   // check to see if it's time to blink the LED; that is, if the difference  
51   // between the current time and last time you blinked the LED is bigger than  
52   // the interval at which you want to blink the LED.  
53   unsigned long currentMillis = millis();  
54  
55   if (currentMillis - previousMillis >= interval) {  
56     // save the last time you blinked the LED  
57     previousMillis = currentMillis;  
58  
59     // if the LED is off turn it on and vice-versa:  
60     if (ledState == LOW) {  
61       ledState = HIGH;  
62     } else {  
63       ledState = LOW;  
64     }  
65  
66     // set the LED with the ledState of the variable:  
67     digitalWrite(ledPin, ledState);  
68   }  
69 }  
70 }  
71 }
```

The IDE will compile, or build, the sketch and if there are no errors, will upload it.



```
28 //  
29 // constants won't change. Used here to set a pin number:  
30 const int ledPin = LED_BUILTIN; // the number of the LED pin  
31  
32 // Variables will change:  
33 int ledState = LOW; // ledState used to set the LED  
34  
35 // Generally, you should use "unsigned long" for variables that hold time  
36 // The value will quickly become too large for an int to store  
37 unsigned long previousMillis = 0; // will store last time LED was updated  
38  
39 // constants won't change:  
40 const long interval = 1000; // interval at which to blink (milliseconds)  
41  
42 void setup() {  
43   // set the digital pin as output:  
44   pinMode(ledPin, OUTPUT);  
45 }  
46  
47 void loop() {  
48   // here is where you'd put code that needs to be running all the time.  
49  
50   // check to see if it's time to blink the LED; that is, if the difference  
51   // between the current time and last time you blinked the LED is bigger than  
52   // the interval at which you want to blink the LED.  
53   unsigned long currentMillis = millis();  
54  
55   if (currentMillis - previousMillis >= interval) {  
56     // save the last time you blinked the LED  
57     previousMillis = currentMillis;  
58  
59     // if the LED is off turn it on and vice-versa:  
60     if (ledState == LOW) {  
61       ledState = HIGH;  
62     } else {  
63       ledState = LOW;  
64     }  
65  
66     // set the LED with the ledState of the variable:  
67     digitalWrite(ledPin, ledState);  
68   }  
69 }  
70 }  
71 }
```

Manual Process if the drivers fail to automatically install.

1. Open Control Panel > Systems & Security > System > Device Manager
2. Look under "Other Devices" for an "Unknown Device"
3. Right click on "Unknown Device" and choose "Update Driver Software" option
4. Select "Browse my computer for the Driver software" option
5. Navigate to the download location of the Arduino IDE
6. You will find a file named "Arduino.inf" located in the "Drivers" folder
7. Windows will complete the install from here

