Lesson 2 Blink

Overview

In this lesson, you will learn how to program your UNO R3 controller board to blink the Arduino's built-in LED, and how to download programs by basic steps.

Component Required:

(1) x Elegoo Uno R3

Principle

The UNO R3 board has rows of connectors along both sides that are used to connect to several electronic devices and plug-in 'shields' that extends its capability.

It also has a single LED that you can control from your sketches. This LED is built onto the UNO R3 board and is often referred to as the 'L' LED as this is how it is labeled on the board.
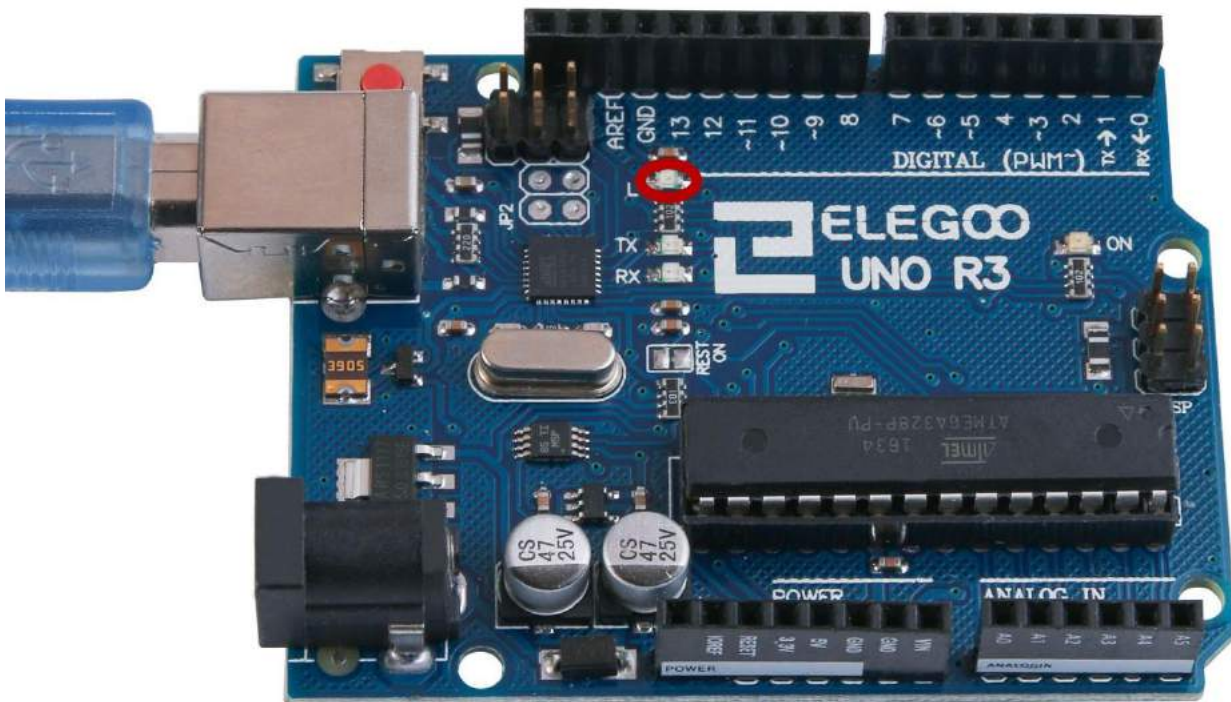
You may find that your UNO R3 board's 'L' LED already blinks when you connect it to a USB plug. This is because the boards are generally shipped with the 'Blink' sketch pre-installed.

In this lesson, we will reprogram the UNO R3 board with our own Blink sketch and then change the rate at which it blinks.

In Lesson 0, you set up your Arduino IDE and made sure that you could find the right serial port for it to connect to your UNO R3 board. The time has now come to put that connection to the test and program your UNO R3 board.

The Arduino IDE includes a large collection of example sketches that you can load up and use. This includes an example sketch for making the 'L' LED blink.

Load the 'Blink' sketch that you will find in the IDE's menu system under File > Examples > 01.Basics
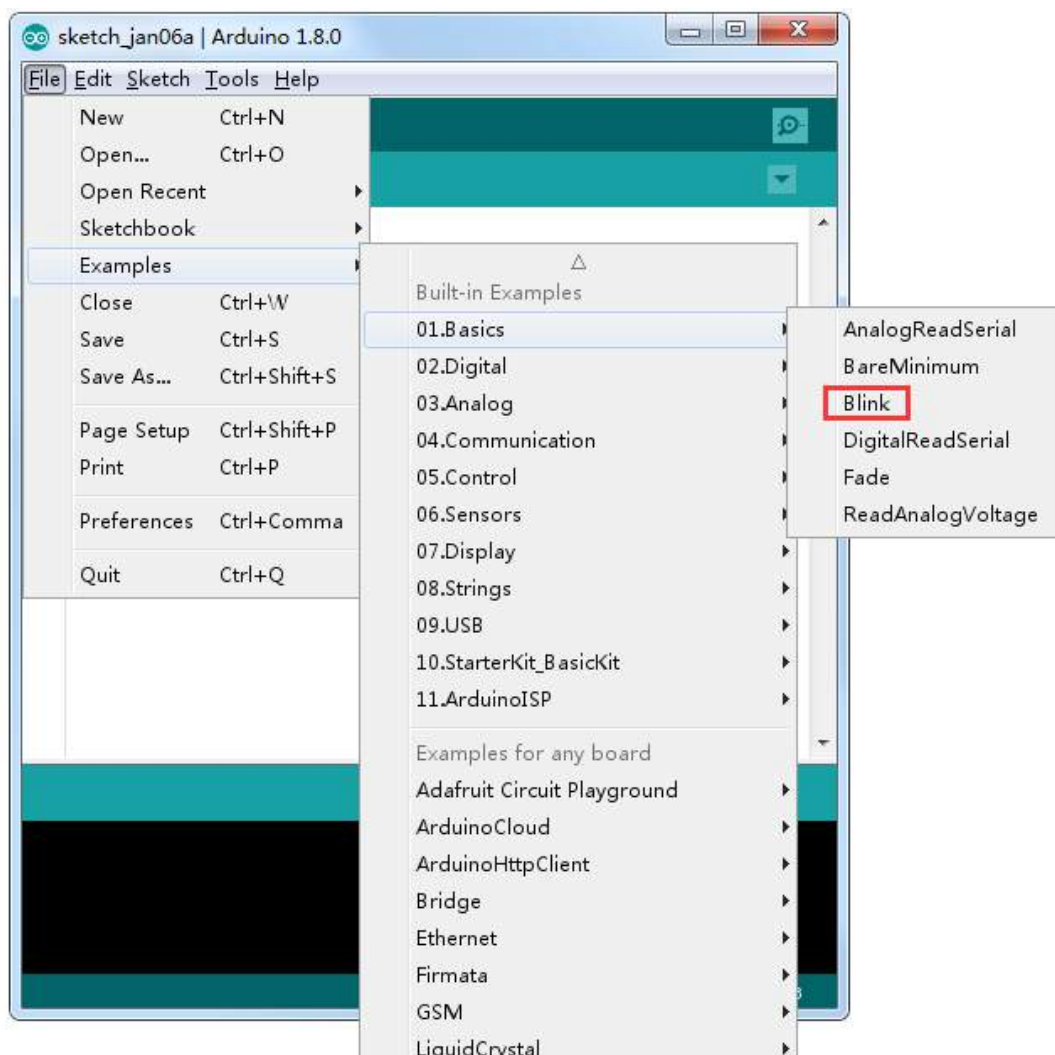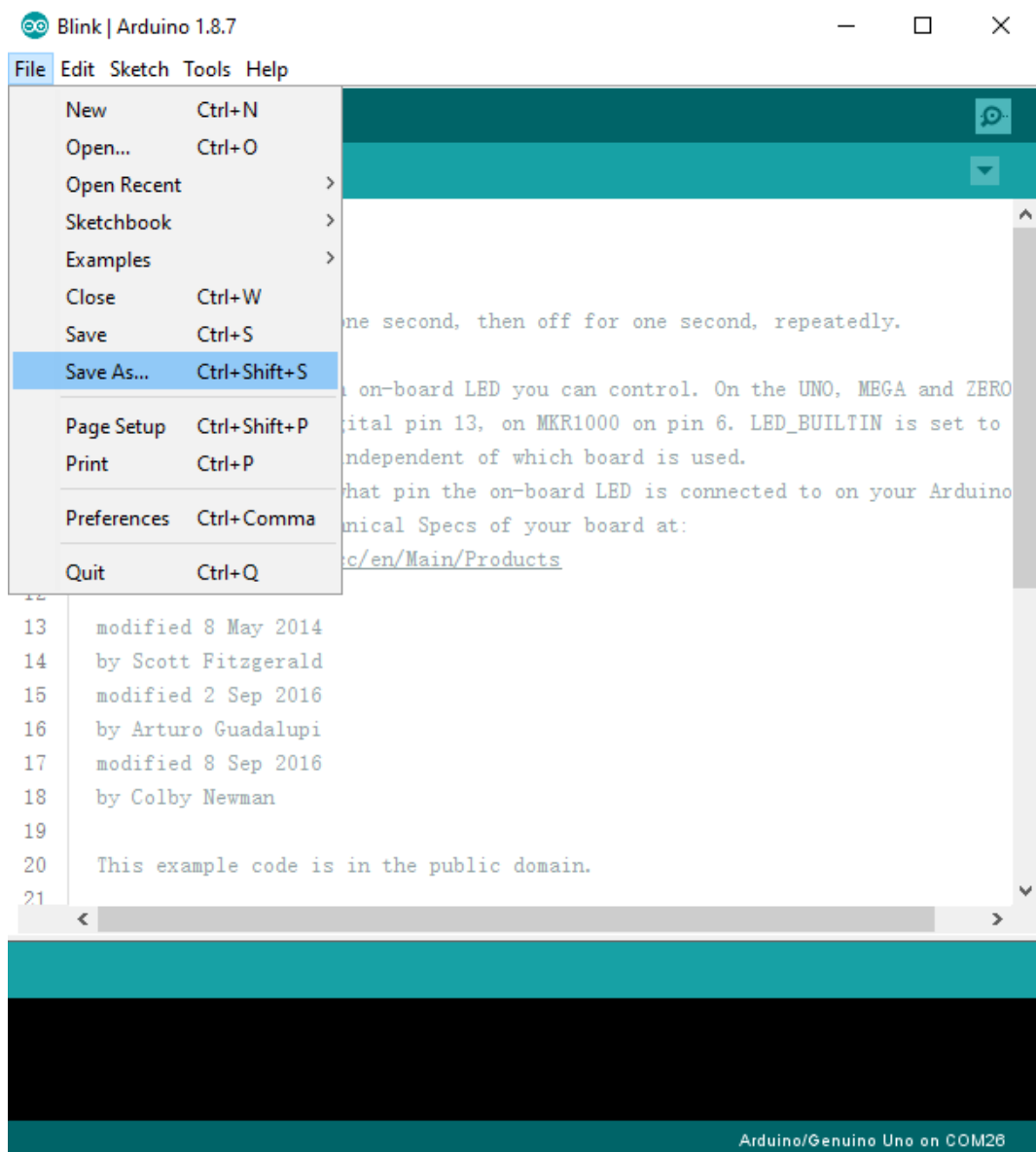
When the sketch window opens, enlarge it so that you can see the entire sketch in the window.
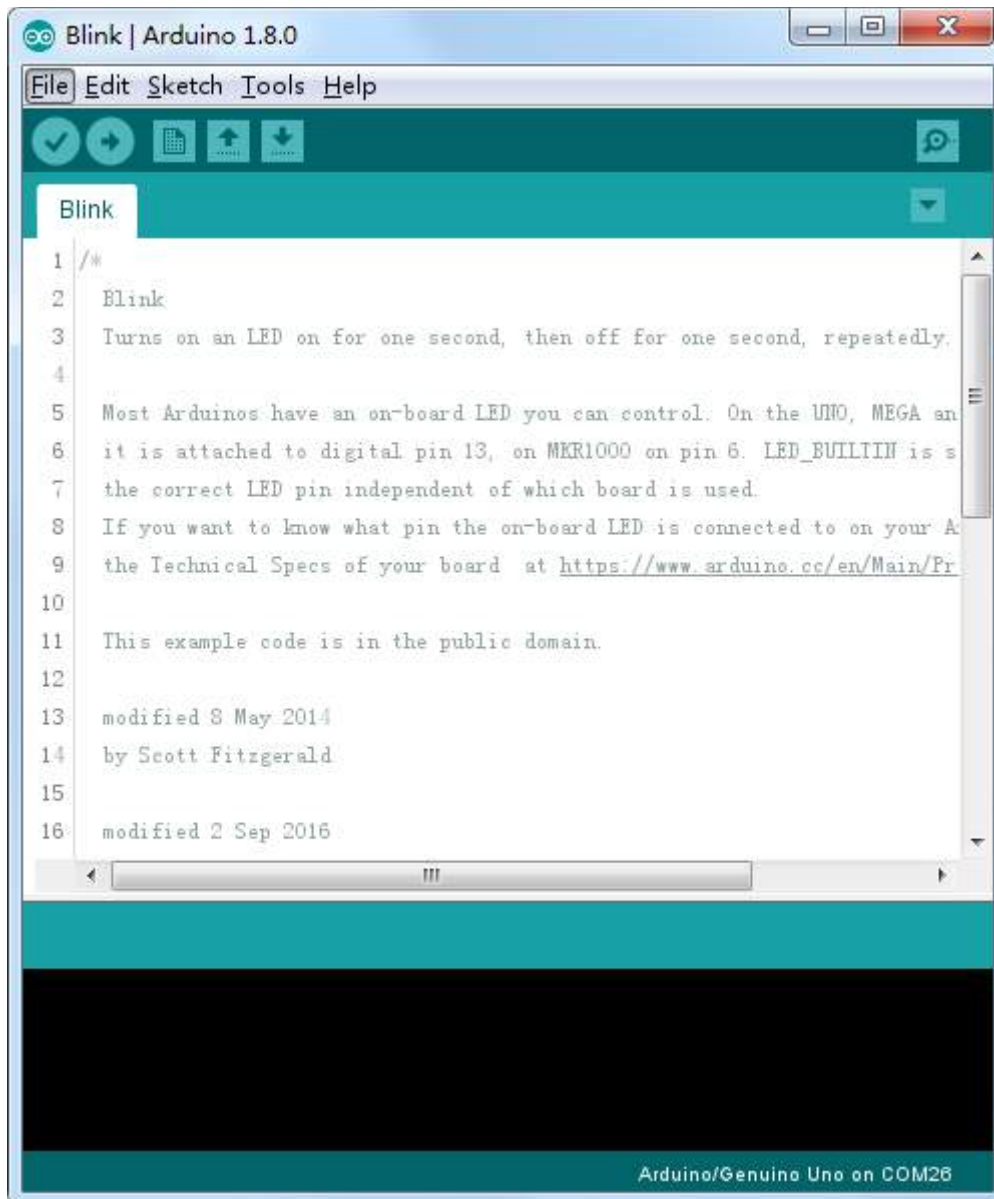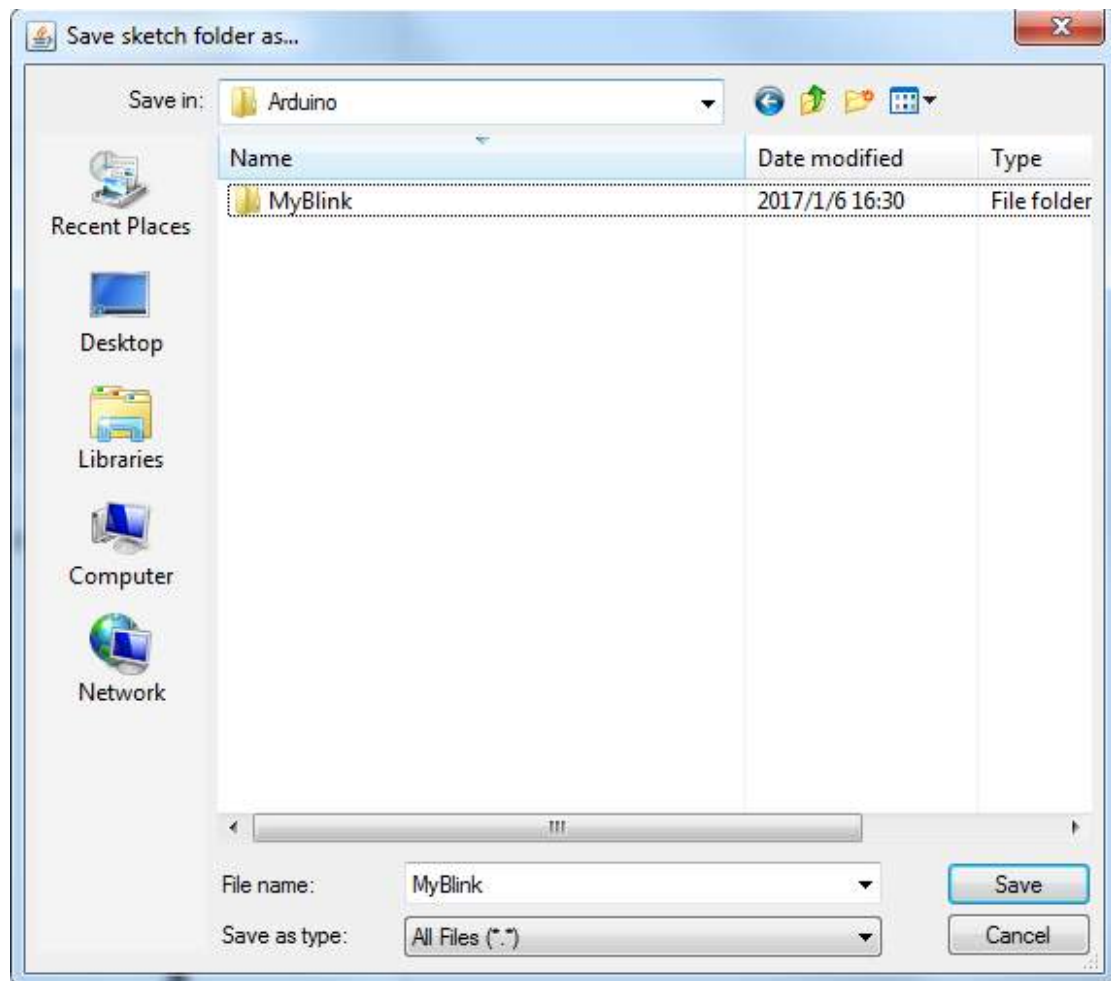

The example sketches included with the Arduino IDE are 'read-only'. That is, you can upload them to an UNO R3 board, but if you change them, you cannot save them as the same file.

Since we are going to change this sketch, the first thing you need to do is save your own copy.

From the File menu on the Arduino IDE, select 'Save As..' and then save the sketch with the name 'MyBlink'.

Blink | Arduino 1.8.7

File Edit Sketch Tools Help

New	Ctrl+N
Open...	Ctrl+O
Open Recent	>
Sketchbook	>
Examples	>
Close	Ctrl+W
Save	Ctrl+S
Save As...	Ctrl+Shift+S
Page Setup	Ctrl+Shift+P
Print	Ctrl+P
Preferences	Ctrl+Comma
Quit	Ctrl+Q

ne second, then off for one second, repeatedly.

on-board LED you can control. On the UNO, MEGA and ZERO
ital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set to
ndependent of which board is used.
hat pin the on-board LED is connected to on your Arduino
nical Specs of your board at:
c/en/Main/Products

```
13     modified 8 May 2014
14     by Scott Fitzgerald
15     modified 2 Sep 2016
16     by Arturo Guadalupi
17     modified 8 Sep 2016
18     by Colby Newman
19
20     This example code is in the public domain.
21
```
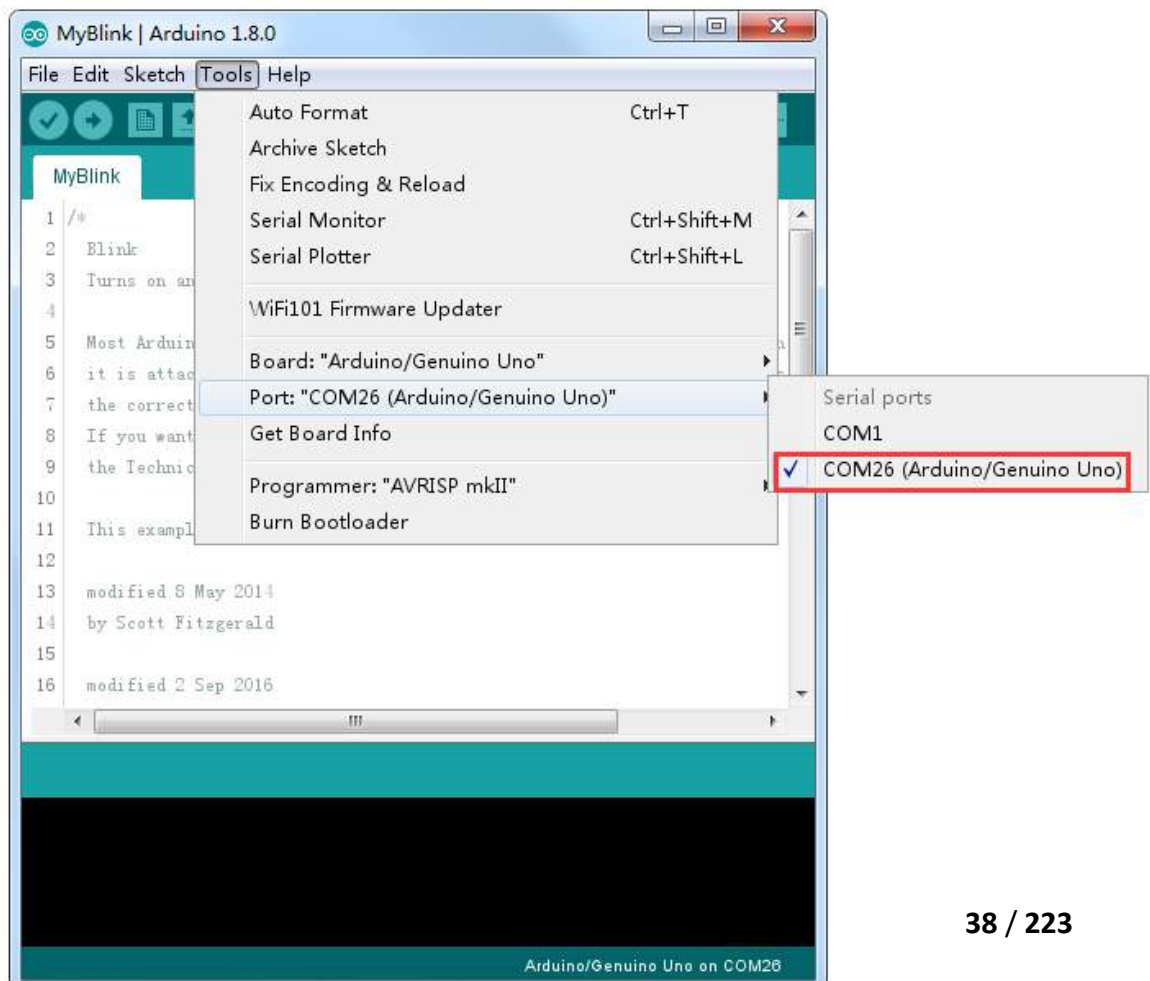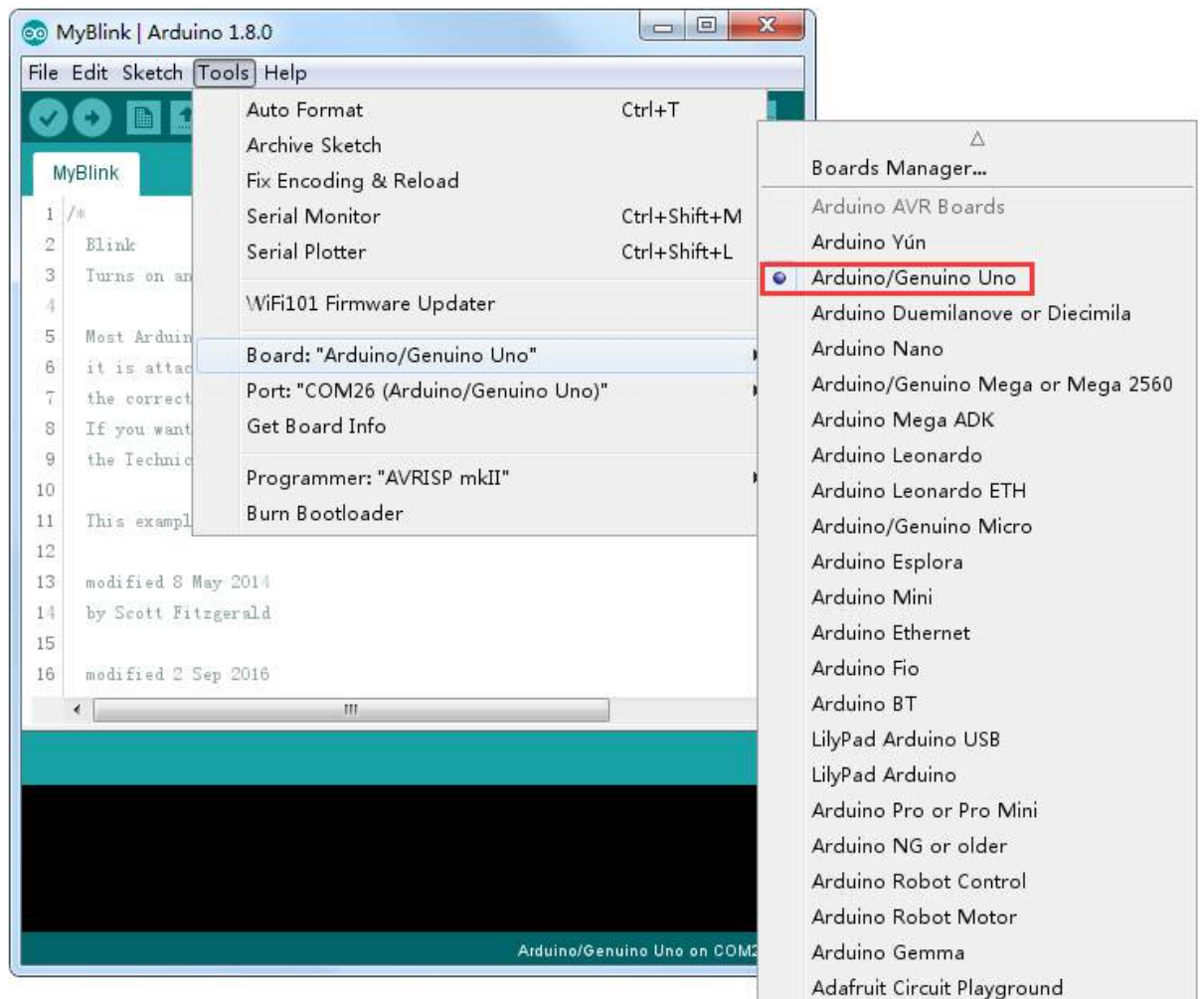
Arduino/Genuino Uno on COM26

You have saved your copy of 'Blink' in your sketchbook. This means that if you ever want to find it again, you can just open it using the File > Sketchbook menu option.
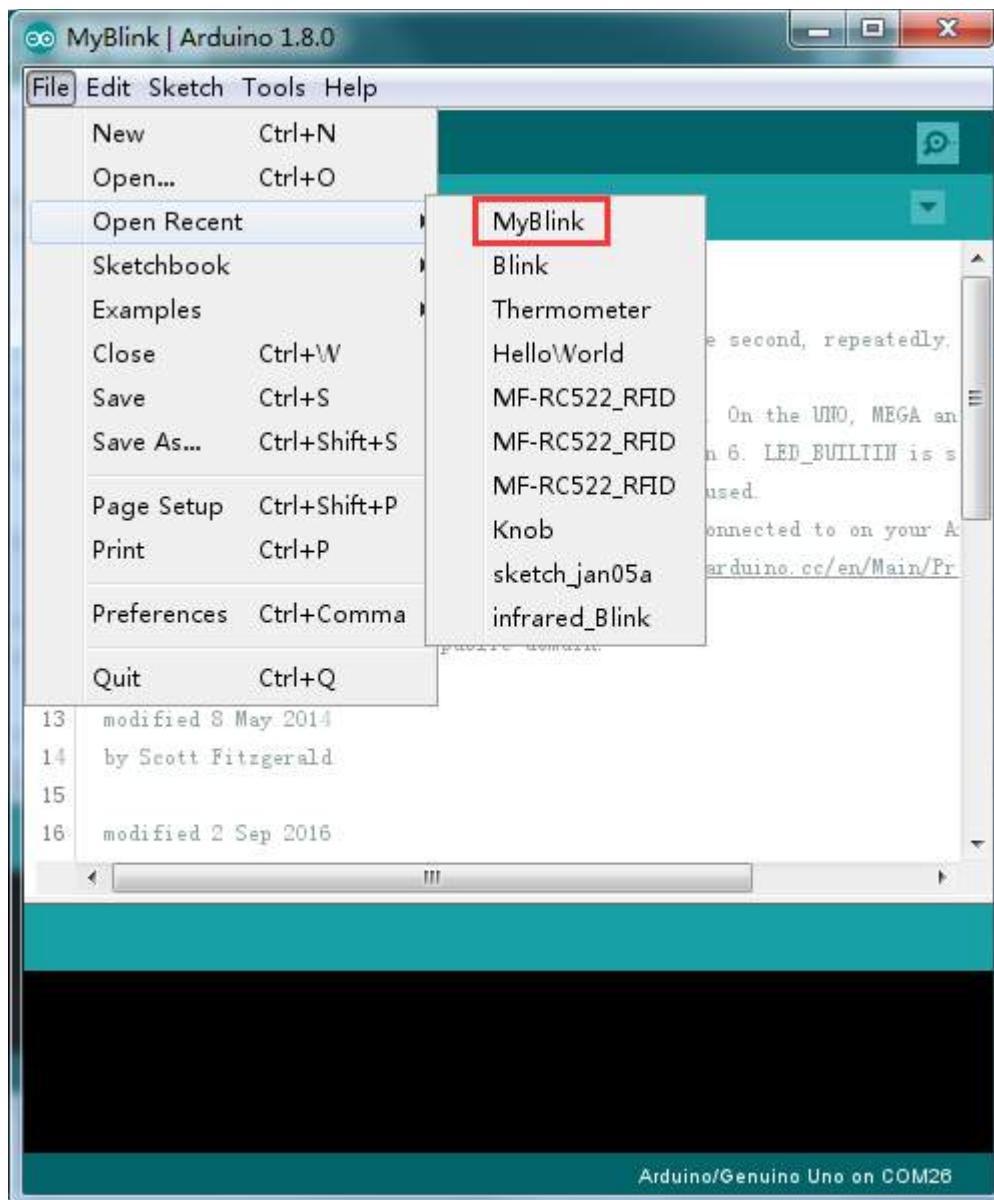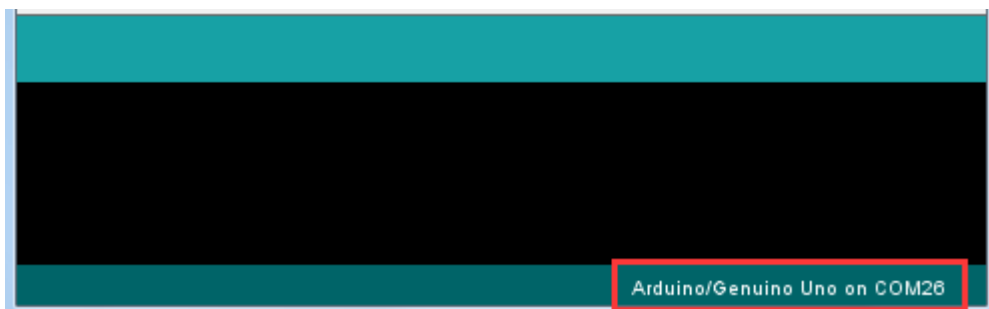
Attach your Arduino board to your computer with the USB cable and check that the 'Board Type' and 'Serial Port' are set correctly.
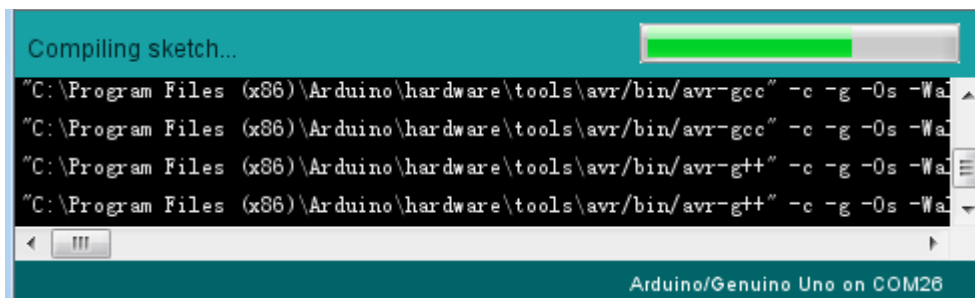
The Arduino IDE will show you the current settings for board at the bottom of the window.

Click on the 'Upload' button. The second button from the left on the toolbar.
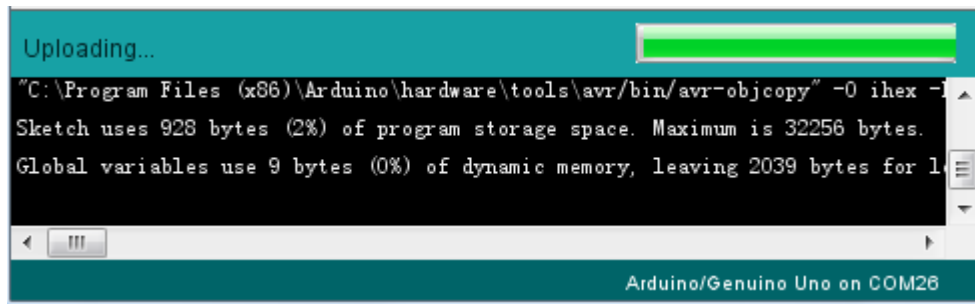

If you watch the status area of the IDE, you will see a progress bar and a series of messages. At first, it will say 'Compiling Sketch...'. This converts the sketch into a format suitable for uploading to the board.
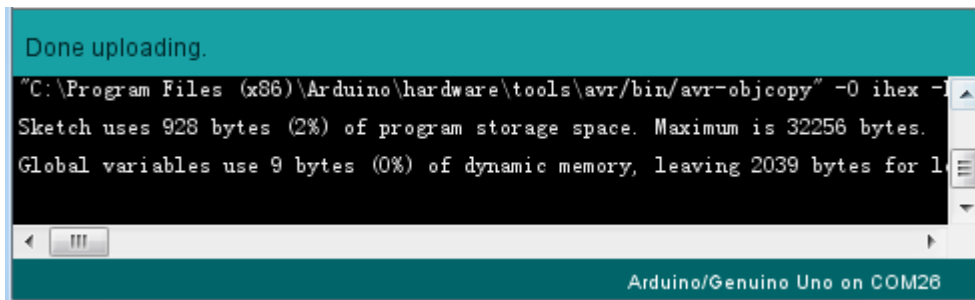

Next, the status will change to 'Uploading'. At this point, the LEDs on the Arduino should start to flicker as the sketch is transferred.

Finally, the staus will change to 'Done'.

The other message tells us that the sketch is using 928 bytes of the 32,256 bytes available.After the 'Compiling Sketch..' stage you could get the following error message:

It can mean that your board is not connected at all, or the drivers have not been installed (if necessary) or that the wrong serial port is selected.

If you encounter this, go back to Lesson 0 and check your installation.

Once the upload has completed, the board should restart and start blinking.

Open the code

Note that a huge part of this sketch is composed of comments. These are not actual program instructions; rather, they just explain how the program works. They are there for your benefit.

The Sketch start with:

```
/*
  Blink
  Turns an LED on for one second, then off for one second, repeatedly.
  ...
  This example code is in the public domain.
  http://www.arduino.cc/en/Tutorial/Blink
*/
// the setup function runs once when you press reset or power the board
```

Those are comments, Anything between the /* and */ is ignored by the Arduino completely, and that is block comment.

Single line comments start with // and everything up until the end of that line is considered a comment.

The first block of code is:

```
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}
```

Every Arduino sketch must have a 'setup' function, and the place where you might want to add instructions of your own is between the { and the }.

In this case, there is just one command there, which, as the comment states tells the Arduino board that we are going to use the LED pin as an output.

It is also mandatory for a sketch to have a 'loop' function.

Unlike the 'setup' function that only runs once, after a reset, the 'loop' function will, after it has finished running its commands, immediately start again.

```
// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

Inside the loop function, the commands first of all turn the LED pin on (HIGH), then 'delay' for 1000 milliseconds (1 second), then turn the LED pin off and pause for another second.

You are now going to make your LED blink faster. As you might have guessed, the key to this lies in changing the parameter in () for the 'delay' command.

```
30 // the loop function runs over and over again forever
31 void loop() {
32   digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the volt
33   delay(500)                         // wait for a second
34   digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the vo
35   delay(500)                         // wait for a second
36 }
```

This delay period is in milliseconds, so if you want the LED to blink twice as fast, change the value from 1000 to 500. This would then pause for half a second each delay rather than a whole second.

Upload the sketch again and you should see the LED start to blink more quickly.