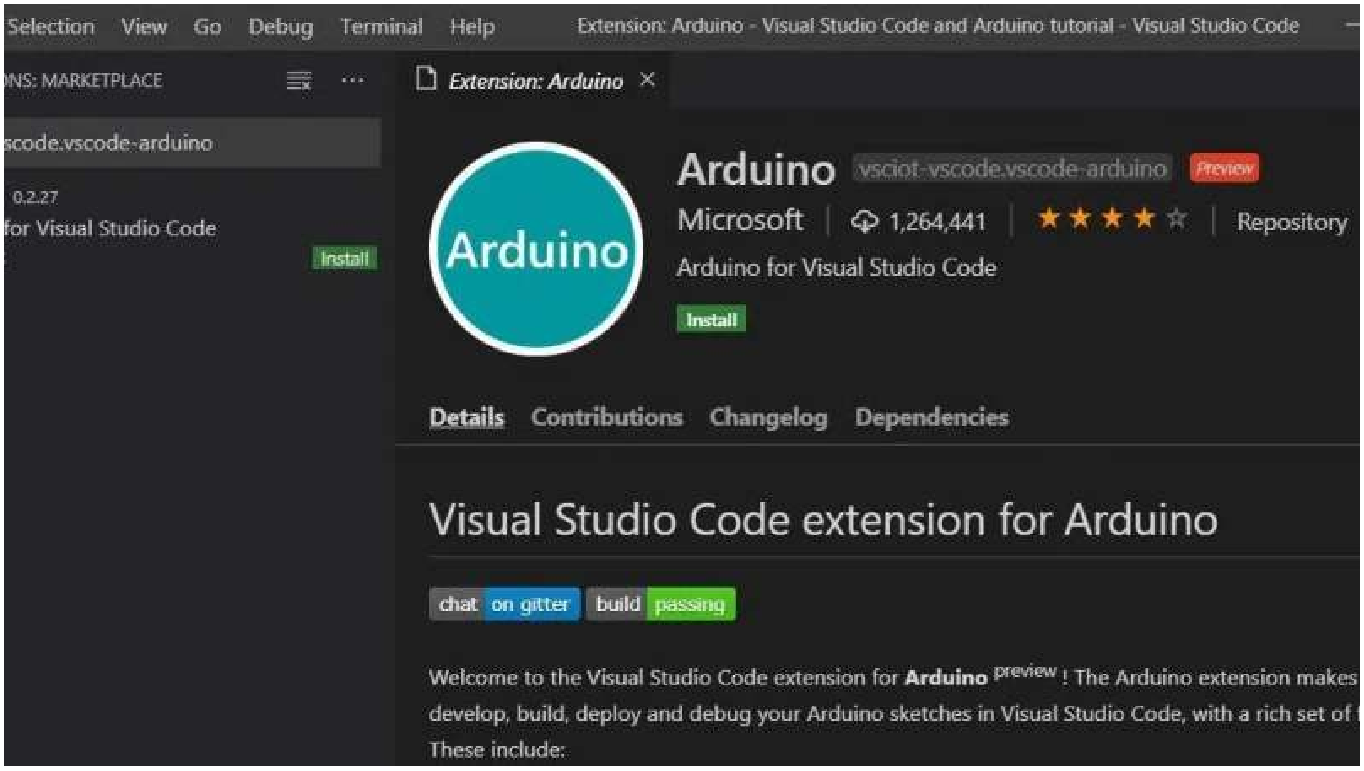
ARDUINO

How to Use Visual Studio Code for Arduino

October 18, 2019 by Paul Lunn

Share


Selection View Go Debug Terminal Help
Extension: Arduino - Visual Studio Code and Arduino tutorial - Visual Studio Code
NS: MARKETPLACE
scode.vscode-arduino
0.2.27
for Visual Studio Code
Install
Extension: Arduino
Arduino
Arduino vsciot-vscode.vscode-arduino Preview
Microsoft | 1,264,441 | ★★★★☆ | Repository
Arduino for Visual Studio Code
Install
Details Contributions Changelog Dependencies
chat on gitter build passing

Visual Studio Code extension for Arduino

Welcome to the Visual Studio Code extension for **Arduino** preview ! The Arduino extension makes
develop, build, deploy and debug your Arduino sketches in Visual Studio Code, with a rich set of
These include:

Learn how you can use the Arduino extension on Visual Studio Code to create programs for your Arduino.

PROJECT

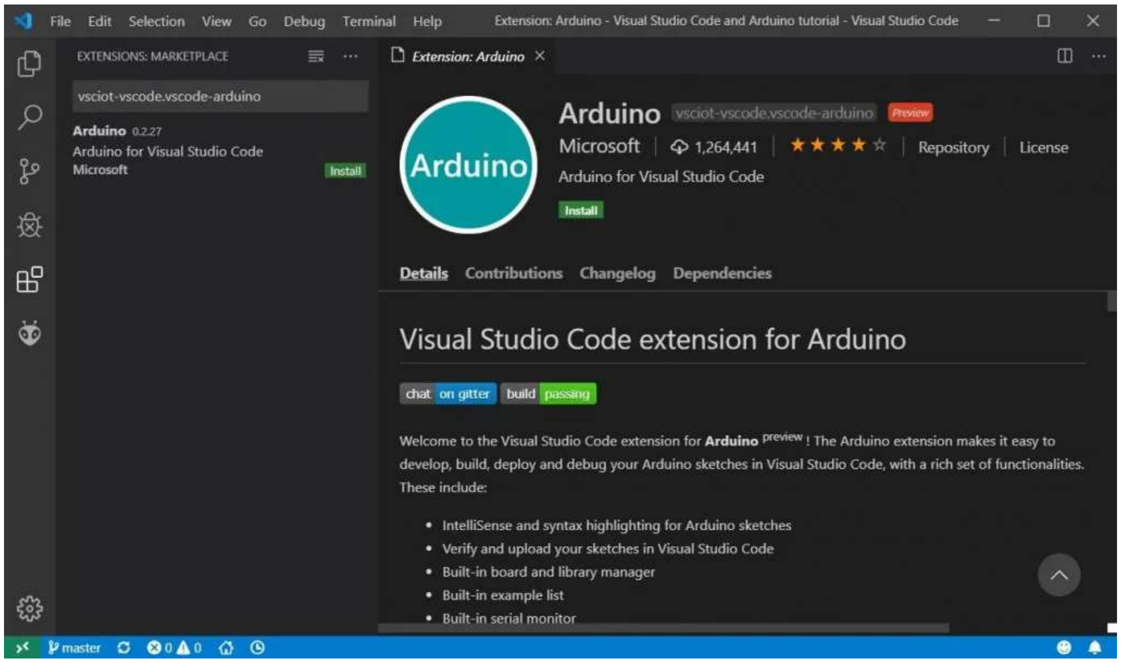
This tutorial is the second in a series of three tutorials looking at software development using Microsoftâ€™s Visual Studio Code (VS Code). If you donâ€™t yet have VS Code installed, head over to

Prerequisites

- VS Code needs to be installed on your system.
- VS Code requires that the standard Arduino IDE is installed, as it uses some of the libraries included in the IDE.

Installation of Arduino Extension

To install the Arduino extension enter â€œCntrl+Shift+xâ€ to display the Extensions viewer and type â€œvscode-arduinoâ€ into the search bar. This should return just one result. Press the â€œInstallâ€ button to install.


File Edit Selection View Go Debug Terminal Help
Extension: Arduino - Visual Studio Code and Arduino tutorial - Visual Studio Code
EXTENSIONS: MARKETPLACE
vsciot-vscode.vscode-arduino
Arduino 0.2.27
Arduino for Visual Studio Code
Microsoft
Install
Extension: Arduino
Arduino
Arduino vsciot-vscode.vscode-arduino Preview
Microsoft | 1,264,441 | ★★★★☆ | Repository | License
Arduino for Visual Studio Code
Install
Details Contributions Changelog Dependencies
chat on gitter build passing
master 0 0 0

Visual Studio Code extension for Arduino

Welcome to the Visual Studio Code extension for **Arduino** preview ! The Arduino extension makes it easy to develop, build, deploy and debug your Arduino sketches in Visual Studio Code, with a rich set of functionalities. These include:

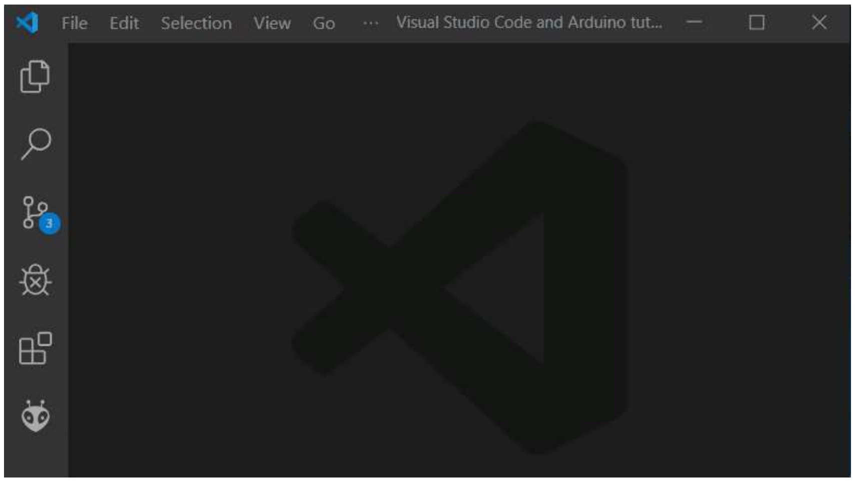
- IntelliSense and syntax highlighting for Arduino sketches
- Verify and upload your sketches in Visual Studio Code
- Built-in board and library manager
- Built-in example list
- Built-in serial monitor

Once the extension is installed, connect an Arduino Uno development board to a USB port of your computer and then restart VS Code â€“ this ensures that everything is installed correctly.

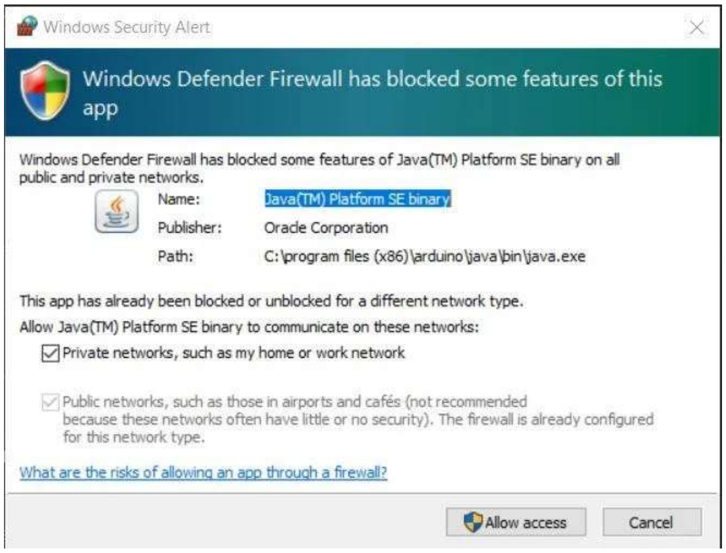
The Command Palette

Not all VS Codeâ€™s functionality is accessible via menus. Instead, VS Code has a command-line-type facility for interfacing with extensions called the Command Palette.

To run the Command Palette, type â€œCntrl+Shift+pâ€ to display a drop-down box that you can enter commands in. Type â€œArduino Examplesâ€ into the command palette and press return. This


File Edit Selection View Go ··· Visual Studio Code and Arduino tut...

When running Arduino commands like those above you may encounter a firewall message like the one shown below. Select â€œAllow Accessâ€ and continue.



Windows Security Alert

Windows Defender Firewall has blocked some features of this app

Windows Defender Firewall has blocked some features of Java(TM) Platform SE binary on all public and private networks.

Name:	Java(TM) Platform SE binary
Publisher:	Oracle Corporation
Path:	C:\program files (x86)\arduino\java\bin\java.exe

This app has already been blocked or unblocked for a different network type.

Allow Java(TM) Platform SE binary to communicate on these networks:

[✓] Private networks, such as my home or work network

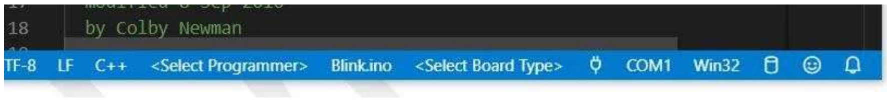
[✓] Public networks, such as those in airports and cafés (not recommended because these networks often have little or no security). The firewall is already configured for this network type.

What are the risks of allowing an app through a firewall?

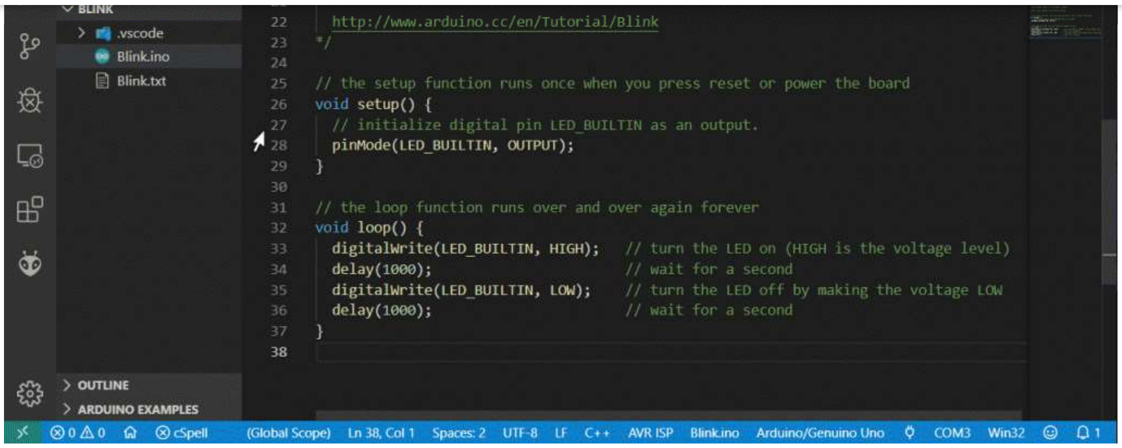
Set up Arduino System

We need to change a few system settings before we can start programming. All of these can be accessed via VS Codeâ€™s interface.

Once you have opened up an Arduino .ino file, VS Code reconfigures in an Arduino mode, and gives access to special functionality in its bottom blue margin, as shown in the image below.


by Colby Newman
TF-8 LF C++ <Select Programmer> Blink.ino <Select Board Type> COM1 Win32

Click on <Select Programmer> and select AVR ISP (Arduino AVR Board)

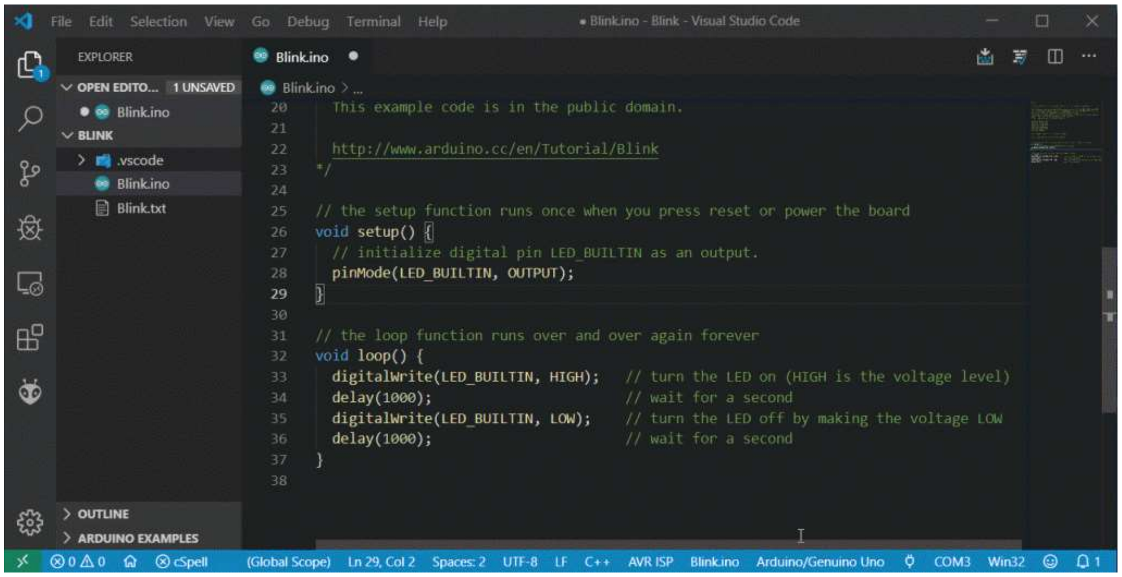


```
  http://www.arduino.cc/en/Tutorial/Blink
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// 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
}
```

Click on <Select Board Type> and select Arduino/Genuino Uno



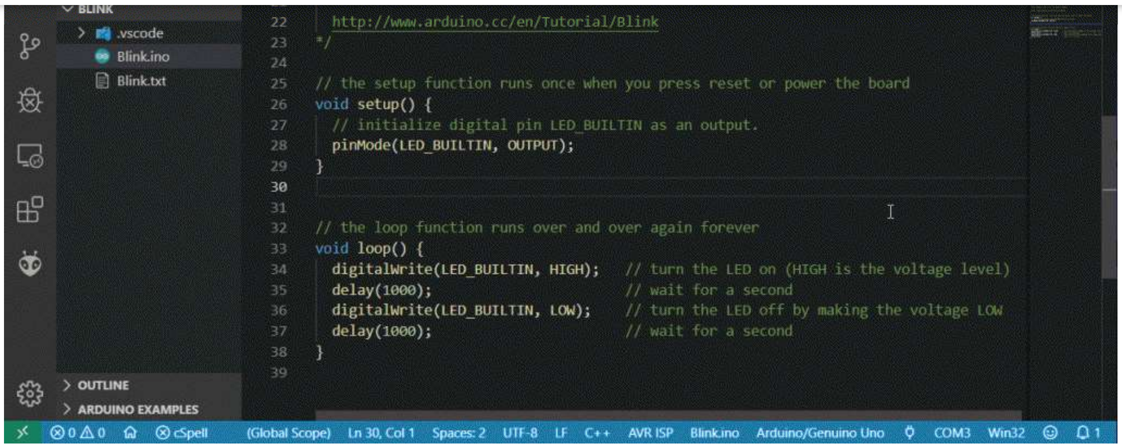
```
  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/Blink
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// 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
}
```

To set the serial port that VS Code will communicate with your Arduino Uno we need to open up the Command Palette (Cntrl+Shift+p) and then type â€œArduino Select Serial Portâ€ . You should then select which USB port your Uno is connected to.

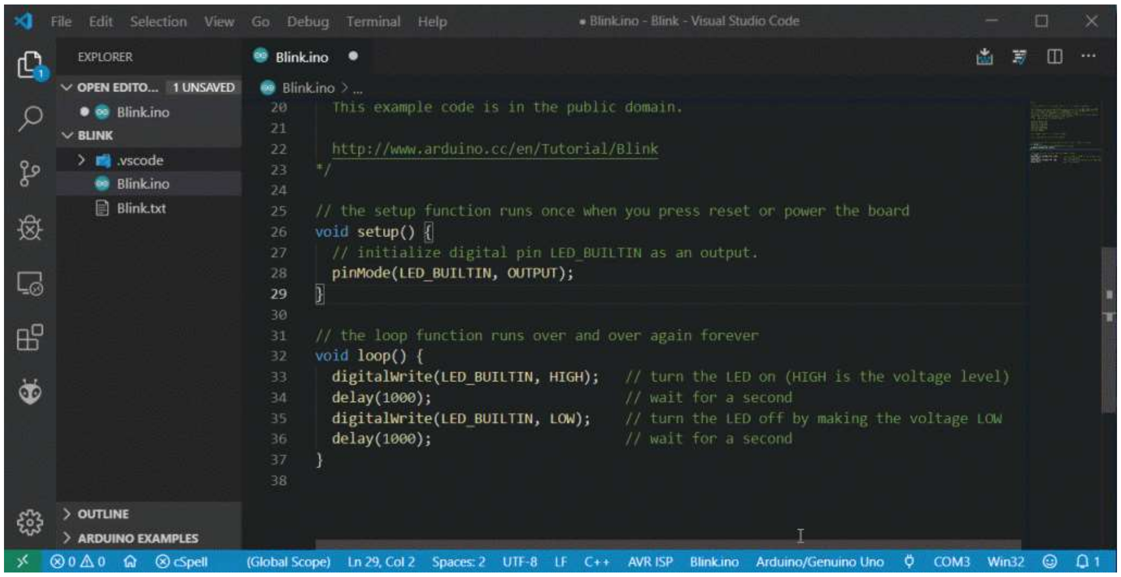


```
  http://www.arduino.cc/en/Tutorial/Blink
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}


// 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
}
```

You can access the Serial Monitor by clicking on the plug symbol on the blue line



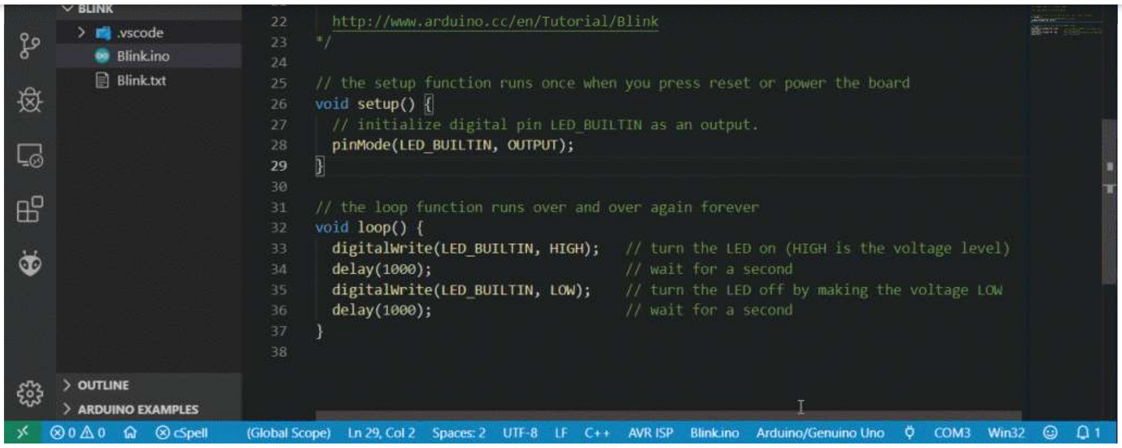
```
  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/Blink
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// 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
}
```

This extension includes a package manager, which makes external package installation really easy. Open up the command palette and type â€œArduino package managerâ€   . A new tab should appear that features many packages to install just by clicking on a button.


BLINK
.vscode
Blink.ino
Blink.txt
OUTLINE
ARDUINO EXAMPLES
⊗ 0 ⚠ 0 ⊗ cSpell (Global Scope) Ln 29, Col 2 Spaces: 2 UTF-8 LF C++ AVR ISP Blink.ino Arduino/Genuino Uno COM3 Win32

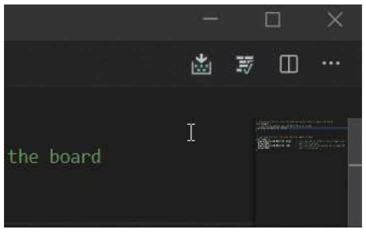
```
  http://www.arduino.cc/en/Tutorial/Blink
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// 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
}
```

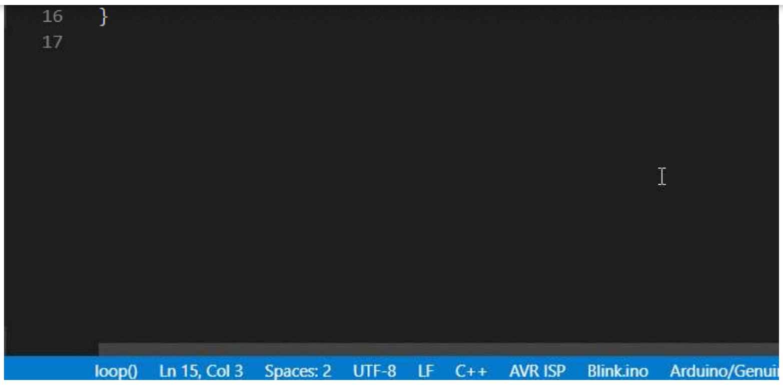
Verify and Upload Your Code

VS Code in Arduino mode provides a couple of convenient icons for you to click for code verification and uploading to your connected Arduino board. These are available in the upper right-hand corner. Either operation will result in a console window appearing at the bottom of VS Code informing you of the outcome.



IntelliSence and Error Checking

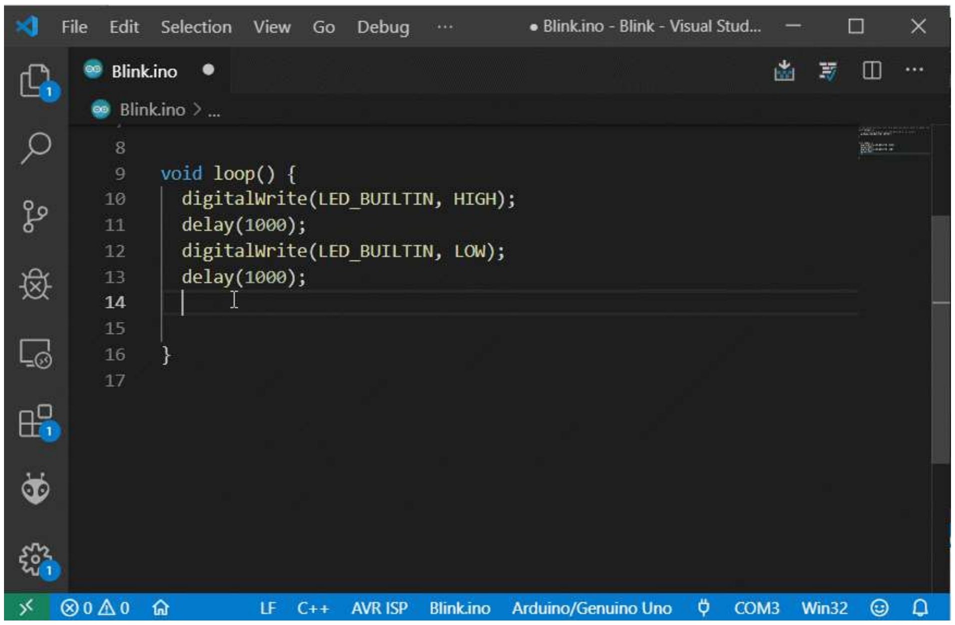
IntelliSence is a code completion engine that looks through all of the source code in your project â€” including external libraries â€” and then suggests functions and variables as you type your code. When you start typing, a drop-down box will appear giving suggestions based upon the word that you are entering. This is an incredibly useful tool that can speed up your coding and also provides a useful way of remembering what functions are available to you.



```
}
```

loop() Ln 15, Col 3 Spaces: 2 UTF-8 LF C++ AVR ISP Blink.ino Arduino/Genui

Software bugs are always a part of programming. VS Code constantly checks your code as you write. If it detects an error, it underlines it with a red squiggle and offers advice. It also keeps a record of the number of current errors in your project in the bottom right-hand corner of VS Code. Clicking on this provides a summary of errors in your code. Double-clicking on an error notification will take you to the error in your code.



File Edit Selection View Go Debug ···
● Blink.ino - Blink - Visual Stud...
Blink.ino
Blink.ino > ...
⊗ 0 ⚠ 0
LF C++ AVR ISP Blink.ino Arduino/Genuino Uno COM3 Win32

```

void loop() {
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000);
  

}

```

Author

Paul Lunn

Dr Paul Lunn is a Lecturer in IOT at Coventry University, UK. He enjoys tinkering with music, programming, Arduino, ESP32, and Raspberry Pi's

Related Content