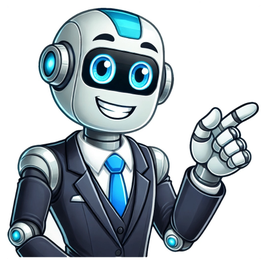
Click to verify



The Arduino Integrated Development Environment (IDE) is a crucial tool for programming and developing projects with Arduino boards. This tutorial will walk you through the steps to download and install the Arduino IDE on your Windows, macOS, or Linux system.1. Visit the Arduino Website: Hover over the 'Software' tab and click on 'Downloads'.2.
Select the Correct Version:Choose the version that matches your operating system (Windows, macOS, or Linux).You can also opt for a portable installation using the ZIP file or the MSI installer for more control.1. Download the Installer:Click on the 'Windows installer' link to download the executable (.exe) file.Double-click the downloaded file to start
the installation process.Read and agree to the license agreement.Select the components you want to install. If unsure, keep all options checked.5. Choose the Installation Path:Select the directory where you want to install the Arduino IDE.6. Finish the Installation:Wait for the installation to complete.Click 'Close' to finish the setup wizard.1. Download
the Installer:Click on the 'macOS' link to download the disk image (.dmg) file.Double-click the downloaded file to open the disk image.3. Drag and Drop to Applications:Drag the Arduino IDE application to the Applications folder.Launch the Arduino IDE from the Applications folder or use Spotlight to search for it.1. Download the Installer:Click on the
'Linux' link to download the AppImage file.2. To install the Arduino IDE on a Linux machine, simply extract the downloaded file to a suitable folder. 3. Now, go to the folder where you extracted the contents to, and look for the arduino-ide file, which will launch the IDE.4. You can also run the file directly in the terminal, navigate to the folder, right
click on it and select run in terminal.5. To launch the editor through the terminal, simply type:Ensure you have the necessary permissions to install software on your system.If you encounter issues, refer to the troubleshooting guides on the Arduino website or forums.Launch and Test the Arduino IDE Find the Arduino IDE in your Start Menu
(Windows) or Applications folder (macOS). Double-click the icon to launch the IDE.Open the Arduino IDE and create a new project to test the installation.3. Now that you have successfully installed the Arduino IDE, you can start creating and running your projects.By following these steps, you should now have the Arduino IDE installed and ready to
use on your system.Downloading and Installing Arduino IDE is very easy. Arduino IDE is a free softwareFirst, go to www.arduino.cc by using a browser. Form the 'SOFTWARE' option click 'DOWNLOADS'. Or simply go to www.arduino.cc/en/Main/Software.From 'Download the Arduino IDE' section select according to your computer operating
system.From 'Contribute to the Arduino Software' window just click on the 'Just Download'.After opening click on 'Agree'.Your Arduino IDE is installed.In this user guide, we will control the output of the ESP32 using the Blynk application installed in our smartphones. Previously, we controlled the ESP modules outputs through applications such as
Telegram and Google Firebase. You can view the tutorials by accessing the links below:Telegram ESP32 and ESP8266: Control GPIOs and LEDs using Arduino IDEGoogle Firebase: Control ESP32 Outputs with Arduino IDEThis time we will show you another useful application through which we will be able to control any output connected with the
ESP32s GPIO pins. Blynk is a free-to-use app where the user can build software to control microcontrollers such as Raspberry Pi, Arduino, and ESP8266 NodeMCU, etc. For the user, it becomes extremely interactive and easy to use this application to build IoT projects from anywhere over the internet. Only the Blynk application and a steady internet
connection in your device (smartphone, laptop, tablet, etc.) are a requirement.For this article, we will keep it simple and control an LED connected to an output GPIO of our ESP32 module through the app in Arduino IDE. The LED will be toggled through two-way communication between the ESP32 board and the Blynk project which you will create
through the application. After that controlling any other ESP32 modules output will be relatively easy to follow. We can also control the buzzer, relay, motors, etc.This article is divided into these sub-topics: Introduction of Blynk app Installing the Blynk app and creating our project Setting up Arduino IDE Controlling LED using ESP32, Arduino IDE
and Blynk application.Blynk is a highly accessible smartphone-based application available for both Android and iOS operating systems. It provides an interactive dashboard where the user can build their own IoT projects by placing components. This will as a result create a mobile app through which the user will be able to control microcontrollers
connected through WIFI. You can control LEDs, relays, electric motors and many more. No internal programming is required to build your project in Blynk. You only need to drag and place your electronic components and it is as easy as that. Thus, building your IoT projects through Blynk is extremely easy and requires very little effort.We will control
a single LED output through the Blynk mobile application. Our aim is to control outputs so we will use an LED for simplicity and ease of use. We will require the following components for this project.ESP32 development boardOne 5mm LEDOne 220ohm resistorBreadboardConnecting WiresFollow the schematic diagram below to assemble your
circuit.Schematic Diagram for the projectWe have connected the anode pin of the LED with GPIO25. Later on, in the program code, we will configure this GPIO pin as an output pin. The cathode pin is grounded through the 220ohm resistor.We will use an Android smartphone for this project. So go to Google Play or Apple Store (if using an
iPhone).Search for Blynk and install the application.You will have to create an account to proceed further. You can also log in with your Facebook account if you possess that.After you have successfully signed in the following window will appear. Click New Project.Now specify the name of your project, device and connection type. After that press the
Create button.You will receive a notification regarding your Authorization token. This can be accessed from the email account you signed in with and also through Project Settings.Now the Blynk canvas will open as shown below:Press on the screen. The widget box will appear. Now we will build our project. Firstly, we will press the Button widget.
You can view information regarding it by pressing the icon present in the far right as highlighted below.Press the button widget again on the canvas to change its parameters. In the output section, choose the GPIO pin through which your LED will be connected to the ESP32 module. We have used GPIO25. You can use any other suitable output GPIO
pin. Also, change the mode of the button to a SWITCH.After you alter the settings of the button the canvas will look something like this:At the top, you will find three icons. These are for Project settings, editing and play button.Inside the play button, you can also find the stop button.The Project Settings show you the settings you applied for your
particular project and a link to the authorization key. The edit button takes you to the widget box. Lastly, the play button lets you connect your project with your microcontroller. Once you press this button you no longer will be able to add new widgets to your project unless you use the STOP button first.As we are building a fairly simple project thus,
we do not need any other widgets besides the button. Our Blynk project is ready. Let us proceed with the Arduino IDE setting.We will use Arduino IDE to program our ESP32 development board. Thus, you should have the latest version of Arduino IDE. Additionally, you also need to install the ESP32 plugin. If your IDE does not have the plugin installed
you can visit the link below:Installing ESP32 library in Arduino IDE and upload code.To use the Blynk app with our ESP32 board, we would have to install its library.To download the library, click here. Click on Code and then Download Zip.You have to go to Sketch > Include Library > Add .zip Library inside the IDE to add the library as well. After
installation of the library, restart your IDE.Open your Arduino IDE and go to File > New to open a new file. Copy the code given below in that file. You need to enter your network credentials. Additionally, you also have to provide your Authorization key which you obtained from the Blynk app.#define BLYNK_PRINT Serial#include #include #include
int led_gpio = 2;#define Authorization_key "hXBwHPyX-kvDSQAxkEq_R_A21x******" #define SSID "Write_your_SSID_here" // replace with your SSID#define Password "********" //replace with your passwordvoid setup() { pinMode(led_gpio, OUTPUT); Serial.begin(115200); delay(10); WiFi.begin(SSID, Password); while (WiFi.status() !=
WL_CONNECTED) { delay(500); Serial.print("."); } Blynk.begin(Authorization_key,SSID,Password);}void loop(){ Blynk.run();}Firstly, we will include the necessary libraries required for this project. WiFi.h will help in establishing the connection between the ESP32 module to a wireless network. We will also include the libraries, for the Blynk
application functionality.#define BLYNK_PRINT Serial#include #include #include Then, we will define the variable of data type int called led_gpio. It will save the GPIO pin through which we will connect our LED. We are using GPIO25 in our case. Make sure to enter the same GPIO pin in the program code which you saved when creating the button
in the Blynk app. Thus, we will control this pin through the app.int led_gpio = 25;Thirdly, we will define the authorization key which was emailed to us through Blynk in our account through which we logged in it. Keep this key safe with you for security concerns.#define Authorization_key "hXBwHPyX-kvDSQAxkEq_R_A21x******" Next, we will define
the network credentials. One for the SSID and the other for the password. These will be our network credentials which will be used to connect to our wireless network. Replace both of them with your credentials to ensure a successful connection.#define SSID "Write_your_SSID_here" #define Password "Write_your_password_here"Inside the setup()
function, we will open a serial connection at a baud rate of 115200.Serial.begin(115200);By using the pinMode() function, the led_gpio variable (GPIO25) will be passed as a parameter inside the function. This will configure the pin as an output.pinMode(led_gpio, OUTPUT); The following section of code will connect our ESP32 board with the local
network whose network credentials we already specified above. We will use the WiFi.begin() function. The arguments will be the SSID and the password which we defined earlier in the code.WiFi.begin(SSID, Password); while (WiFi.status() != WL_CONNECTED) { delay(500); Serial.print("."); }Now, we will connect our ESP32 module with the Blynk
app by using the Blynk.begin() function. It takes in three parameters. The first is the authorization key. The second is the SSID and the third parameter is the password of the WIFI. All three of these were defined earlier on in the program code. Through correct credentials, a successful connection will be
made.Blynk.begin(Authorization_key,SSID,Password);Inside the loop function we will use the Blynk.run() command to keep the connection running.Blynk.run();Choose the correct board and COM port before uploading your code to the board.Go to Tools > Board and select ESP32 Dev Module.Next, go to Tools > Port and select the appropriate port
through which your board is connected.Click on the upload button to upload the code to your ESP32 development board.After you have uploaded your code to the development board, press its ENABLE button.Press ENABLE ButtonIn your Arduino IDE, open up the serial monitor and you will be able to see the status of your WIFI connection. You will
receive different messages as shown below.Serial MonitorWhen you receive the ready message on the serial monitor, open your Blynk application. Open the project you created and press the play button.Press Play ButtonClick the microcontroller icon at the top. You will receive a notification about the connection being established with your ESP32
board.Now, touch the Button widget to toggle the LED. By pressing ON/OFF, simultaneously the LED connected to GPIO25 will turn ON/OFF as well. To disrupt the connection, press the stop button.Press ON/OFF Button to toggle LEDIn conclusion, we were able to control our ESP32 modules output by creating a mobile app in Blynk. We controlled
an LED connected to a pin configured as an output. Similarly, you can use any other output GPIOs easily and control it through the app.Other ESP32 Projects:Learn about the IDEs, Web Editor, CLI and all the software tools that you need to get your hands dirty.Discover all the features of the Arduino IDE, our most popular programming tool.Arduino
CLI is a multifunctional tool with Boards/Library Managers, sketch builder for Arduino-compatible boards via command line.The Arduino solution to program PLC compatible Arduino devices using IEC 61131-3 standard languages (Windows only). Downloadable for free.IntroductionInstalling the Arduino Integrated Development Environment (IDE) on
your PC is a straightforward process that allows you to start programming Arduino boards. In this blog post, we will walk you through the steps needed to get the IDE up and running on your computer, along with guidance on how to troubleshoot common issues.Step 1: Download the Arduino IDEThe first step in the installation process is to download
the Arduino IDE. Visit the official Arduino website at arduino.cc/en/software. On this page, you will find different options for downloading the IDE based on your operating system, which includes Windows, macOS, and Linux. Ensure you select the correct version for your system. After clicking the download link, the file will be saved to your system,
typically in the 'Downloads' folder.Step 2: Install the Arduino IDEOnce the download is complete, locate the downloaded file on your computer and double-click it to start the installation process. Follow the prompts in the setup wizard. For Windows, you may need to authorize the installation through User Account Control (UAC). Make sure to agree
with the terms and conditions, and choose the default installation location unless you have specific requirements to change it. Accept the default options for installation components to install the essential tools required for programming.Step 3: Launch the Arduino IDEAfter the installation is completed, you can launch the Arduino IDE either from the
installation wizard by selecting the 'Launch' option or by finding it in your start menu or applications folder. The interface may take a moment to load as it initializes all necessary components. Once up, youll see a sketchbook window; this is where you will write, modify, and upload your code.Step 4: Set Up the Arduino BoardTo begin using the
Arduino IDE effectively, you need to connect your Arduino board to your PC using a USB cable. Once connected, open the IDE and navigate to 'Tools' in the top menu. Here, select 'Board' and choose the model of your Arduino board from the list. Then, select 'Port' and choose the corresponding COM port that matches your connected board. This step
ensures that the IDE can communicate with your hardware.Step 5: Test the InstallationTo confirm that everything is working correctly, load a sample sketch. Go to 'File' > 'Examples' and choose 'Basics' > 'Blink'. This simple program will make the onboard LED on your Arduino board blink. Click the upload button (right arrow) in the IDE to transfer
the sketch to your board. If the LED blinks, congratulations! Youve successfully installed and tested the Arduino IDE on your PC.ConclusionInstalling the Arduino IDE is the first step towards diving into the world of electronics and programming. By following these simple steps, you can set up your IDE without any hassle. Happy coding!In this tutorial,
you will learn how to assign a fixed or static IP address to ESP32. To assign a static IP address to ESP32, we need to set local IP, gateway, subnet, primary and secondary DNS. You must be wondering what is meant by assigning a static or fixed IP address to EP32. When you create a web server with ESP32, we always access the web server through
an IP address. But whenever we restart or reboot ESP32, we possibly get a new IP address. WiFi network usually assigns a new IP address to ESP32 board. Therefore, we can assign a static or fix IP address to our board. Shortly, you will be able to do fix the IP address of ESP32 with a very easy method.An IP address, short for Internet Protocol
address, is a numerical label assigned to each device connected to your router for Internet connectivity. It serves two main purposes: identifying the host or network interface and providing the location of the device in the network. By using an IP address, devices can send and receive data over the Internet, enabling communication between
computers, servers, and other networked devices worldwide.IP addresses are used to facilitate the routing of data packets across the Internet. They consist of a series of numbers separated by periods (e.g., 192.168.0.1) and are typically represented in either IPv4 (Internet Protocol version 4) or IPv6 (Internet Protocol version 6) format.IP addresses
can be either static or dynamic. A static IP address remains fixed and is manually assigned to a device, whereas a dynamic IP address is automatically assigned by a router using DHCP protocol and may change over time.A DHCP server (Dynamic Host Configuration Protocol server) in a Wi-Fi router serves an essential role in managing and allocating
IP addresses to devices on a local network. The DHCP server assigns IP addresses dynamically to devices that connect to the Wi-Fi network. When a device connects to the network and requests an IP address, the DHCP server assigns an available address from the pool. Along with IP addresses, the DHCP server can provide additional configuration
parameters to devices. These parameters include:Subnet maskDefault gatewayDNS server addressesThe DHCP server pushes these parameters to connected devices, enabling them to configure their network settings automatically.To assign a static IP address to ESP32, we need to configure the above parameter settings on the ESP32 manually and
the DHCP server will not configure those network settings. Hence, to assign a static IP address we must know the above given parameters before connecting ESP32 to a Wi-Fi network.For an illustration of how to assign a static IP address to ESP32, we use the example of an ESP32 Web server with DHT11/DHT22. We will use the same sketch use in
that tutorial except adding the functionality of assigning fix IP address. In order to understand the code given below, I suggest you check the ESP32 Web server with DHT11/DHT22 interfacingtutorial. In this tutorial, I will only explain those functions that are used for assigning fix IP addresses.#include #include #include "DHT.h"// Uncomment one of
the lines below for whatever DHT sensor type you're using!//#define DHTTYPE DHT11 // DHT 11//#define DHTTYPE DHT21 // DHT 21 (AM2301)#define DHTTYPE DHT22 // DHT 22 (AM2302), AM2321//DHT Sensor;uint8_t DHTPin = 4;// Initialize DHT sensor.DHT dht(DHTPin, DHTTYPE);float Temperature;float Humidity;// Replace with your network
credentialsconst char* ssid = "Enter your WiFi name here"; // Enter SSID hereconst char* password = "Enter your WiFi password herer"; //Enter Password here// Set web server port number to 80WiFiServer server(80);// Variable to store the HTTP requestString header;// it wil set the static IP address to 192, 168, 1, 184IPAddress local_IP(192, 168,
1, 184);//it wil set the gateway static IP address to 192, 168, 1,1IPAddress gateway(192, 168, 1, 1);// Following three settings are optionalIPAddress subnet(255, 255, 0, 0);IPAddress primaryDNS(8, 8, 8, 8); IPAddress secondaryDNS(8, 8, 4, 4);void setup() {Serial.begin(115200);pinMode(DHTPin, INPUT);dht.begin();// This part of code will try create
static IP addressif (!WiFi.config(local_IP, gateway, subnet, primaryDNS, secondaryDNS)) {Serial.println("STA Failed to configure");}// Connect to Wi-Fi network with SSID and passwordSerial.print("Connecting to ");Serial.println(ssid);WiFi.begin(ssid, password);while (WiFi.status() != WL_CONNECTED) {delay(500);Serial.print(".");}// Print local IP
address and start web serverSerial.println("");Serial.println("WiFi connected.");Serial.println("IP address: ");Serial.println(WiFi.localIP());server.begin();}void loop(){Temperature = dht.readTemperature(); // Gets the values of the temperatureHumidity = dht.readHumidity(); // Gets the values of the humidity WiFiClient client = server.available(); //
Listen for incoming clientsif (client) { // If a new client connects,String request = client.readStringUntil('\r'); client.println("HTTP/1.1 200 OK");client.println("Content-type:text/html");client.println("Connection: close");client.println();// Display the HTML web
pageclient.println("");client.println("");client.println("");client.println("");client.println("setInterval(loadDoc,200);");client.println("function loadDoc() {");client.println("var xhttp = new XMLHttpRequest();");client.println("xhttp.onreadystatechange = function() {");client.println("if (this.readyState == 4 && this.status == 200)
{");client.println("document.getElementById(\"webpage\").innerHTML =this.responseText}");client.println("};");client.println("xhttp.open(\"GET\", \"/\", true);");client.println("xhttp.send();");client.println("}");client.println(")"); // CSS to style the tableclient.println("body { text-align: center; font-family: \"Arial\", Arial;}");client.println("table { border-
collapse: collapse; width:40%; margin-left:auto; margin-right:auto;border-spacing: 2px;background-color: white;border: 4px solid green; }");client.println("th { padding: 20px; background-color: #008000; color: white; }");client.println("tr { border: 5px solid green; padding: 2px; }");client.println("tr:hover { background-color:yellow;
}");client.println("td { border:4px; padding: 12px; }");client.println(".sensor { color:white; font-weight: bold; background-color: #bcbcbc; padding: 1px; }"); // Web Page Headingclient.println("ESP32 Web Server Reading sensor
values");client.println("DHT11/DHT22");client.println("Microcontrollerslab.com");client.println("MEASUREMENTVALUE");client.println("Temp. Celsius");client.println(dht.readTemperature());client.println(" *C"); client.println("Temp. Fahrenheit");client.println(1.8 * dht.readTemperature() + 32);client.println(" *F");
client.println("Humidity");client.println(dht.readHumidity());client.println(" %"); client.println("");client.println();Serial.println("Client disconnected.");Serial.println("");}}Now copy this code to your Arduino IDE.Before uploading this code to ESP32 board, you need to enter SSID name and Password in these lines that are your WiFi name and
password.// change it with your WiFi use name and password const char* ssid = "Enter your WiFi name here"; // Enter SSID hereconst char* password = "Enter your WiFi password here"; //Enter Password hereAs mentioned earlier, all code is the same as used in DHT11 interfacing with ESP32 tutorial except the code which adds the functionality of
fixing the IP address of ESP32. Now, we will see how to add this functionality.Above the setup() function, these lines are used to assign a static IP address to the ESP32 development board.PAddress local_IP() functions assign the IP address of 192, 168, 10, 47 through a gateway of 192, 168, 2 2.You can choose the IP address and gateway address to
whatever value you want according to the available list of IP addresses in your local area network.Last three lines set the subnet, primary DNS and secondary DNS for network settings. I recommend you do not change these values unless you really understand the concepts of networking.primaryDNS and secondaryDNS functions are optional here.
you can delete them if you want.// it wil set the static IP address to 192, 168, 10, 47IPAddress local_IP(192, 168, 10, 47);//it wil set the gateway static IP address to 192, 168, 2,2IPAddress gateway(192, 168, 2 ,2);// Following three settings are optionalIPAddress subnet(255, 255, 0, 0);IPAddress primaryDNS(8, 8, 8, 8); IPAddress secondaryDNS(8, 8, 4,
4);Inside the setup() function, WiFi.config() function used to fix the IP address according to above given parameters such as local_IP, gateway, subnet, primaryDNS, secondaryDNS.If the IP address is available in your local area network, it will assign this IP address. Otherwise, you will see the message TA Failed to configure on the serial monitor.//
This part of code will try create static IP addressif (!WiFi.config(local_IP, gateway, subnet, primaryDNS, secondaryDNS)) {Serial.println("STA Failed to configure");}Now your ESP32 sketch is ready to be uploaded. Click on the upload button in your Arduino IDE.After uploading the code, open the serial monitor of Arduino IDE. You will see this
message on a serial monitor which shows you the IP address you assigned.This is the same IP address that we mentioned inside the sketch and the local area network has assigned the same IP address to the ESP32 board.Now even if you restart or reboot your ESP32 board, you will always get the same IP address and you can access the ESP32 web
server with the same IP address.This section provides details on functions used to assign static IP.WiFi.config(): This function is used to configure a fixed IP address and also modify the DNS, gateway, and subnet addresses for ESP32 board. WiFi.begin() function automatically assigns an IP address to devices from the available list of IP addresses in
your network.If you call WiFi.config() before WiFi.begin(), it will force the .begin to assign IP address according to define IP address inside .config().Static IP addresses have various applications in different scenarios. Here are some common use cases for static IP addresses:ESP Web Servers: Static IP addresses are commonly used for hosting servers
such as web servers, FTP servers, email servers, or game servers. A static IP address ensures that the server is always reachable at a consistent address, making it easier for users or clients to connect to the services provided by the server.Remote Access: Static IP addresses are useful when you need to remotely access devices or networks. For
example, if you want to set up remote desktop access to a computer or establish a virtual private network (VPN) connection, a static IP address ensures a stable connection without the need to constantly update IP address settings.Networked Devices: Certain networked devices or Internet of Things (IoT) devices may require a static IP address for
configuration and management purposes. This ensures that the devices are consistently accessible on the network, making it easier to interact with or control them.Port Forwarding: Static IP addresses are often required for port forwarding on routers. Port forwarding allows external devices or services to connect to specific ports on a device within a
local network. By using a static IP address, you can reliably configure port forwarding rules since the IP address remains constant.Security and Access Control: Static IP addresses can be used for security and access control purposes. For example, in a firewall or network access control system, specific IP addresses may be whitelisted or granted
special permissions for enhanced security.VPN Services: When setting up a virtual private network (VPN) server or subscribing to a VPN service, a static IP address may be required to ensure consistent access and connectivity.These are just a few examples of how static IP addresses are used in different applications. In general, static IP addresses
are advantageous when consistent and reliable connectivity is needed, or when specific network configurations and services rely on fixed IP addresses.I hope you enjoyed this tutorial, you may also like to check other ESP32 tutorials and projects:Experience the Arduino IDE online. Whether you're at home or on the go, code, upload and access your
projects anytime from your browser for free.Go to Cloud EditorLearn more[ide]Removed log4j from IDE. Fix CVE-2021-45105, CVE-2021-45046ARDUINO 1.8.18 2021.12.14 SECURITY HOTFIX RELEASE[ide]Upgrade log4j to 2.16.0 - Extra hardening for CVE-2021-44228ARDUINO 1.8.17 (not released superseded by 1.8.18)[ide]Upgrade log4j to
2.15.0 - CVE-2021-44228 (thanks @rhowe)ARDUINO 1.8.16 2021.09.06[ide]Removed the very outdated off-line documentation.Do not crash if a malformed boards index URL is entered in preferences.Fixed font-resizing problem with multiple tabs open.Updated FTDI drivers to version 2.12.36.4[wifi-firmware]Added latest firmwares (up to version
1.4.8) for NINA-based boardsARDUINO 1.8.15 2021.05.13HOTFIX:Fixed regression in library discovery (failing to compile bundled libs, like SD or Ethernet, on some platforms)ARDUINO 1.8.14 2021.05.12[ide]Fix status bar custom board preferences disappearing (thanks @magedrifaat)Boards Manager bugfix: sometimes the "Install" button
appeared instead of "Update" after entering a search term.Libraries/Boards Manager: the Type and Category filters are kept after install/uninstall.Added 'deprecated' label support in Boards Manager, platforms marked as deprecated are shown at the bottom of the list.Serial Plotter: line endings dropdown menu now comply with the global preference
settings. (thanks @sceniclife)Libraries Manager: Maintainer is displayed instead of the Author.Firmware Updater: Added new firmwares and support for Nano RP2040 Connect.Boards Manager: cached 3rd party package_index.json are no more deleted if the origin URL is removed from preferences, this allows other tools (like arduino-cli), that may
have different URLs configuration, to work properly.The board configuration submenus are now scrollable.New shortcut: Shift+click on Serial Monitor button will open the Serial Plotter. (thanks @n-elia)Fixed NPE if no board is selected. (thanks @matthijskooijman)[arduino-builder]Improved lib detection: check for matching name in
library.propertiesFix compile if invalid build.options.json is found (thanks @d-a-v)Improved error messages for .cpp/.h file in the sketchARDUINO 1.8.13 2020.06.16[ide]Fixed crash on MacOSX >=10.15 with multiple monitor setups.The boards in the boards list in the "Tools" menu are now grouped by platform. This should make the selection easier
when there are lots of boards loaded in the IDE. (thanks @matthijskooijman)Fixed save of proxy username/password settings in "manual proxy" section.In "Tools / Programmers" menu only shows programmers related to the currently selected board. (thanks @matthijskooijman)Fixed: sketch window out-of-display if the IDE is restarted after removing
an external monitor.Allow baudrate/config change when the Serial Monitor/Plotter is disabled. (thanks @matthijskooijman)Updated jmdns to 3.5.5. Should improve network board detection.Optimized libraries scanning during startup. (thanks @ricardojlrufino)Better vertical centering of file names in tabs. (thanks @ricardojlrufino)Better console
rendering of progress bar during upload. (thanks @matthijskooijman)[arduino-builder]FIX REGRESSION: Fixed precompiled libraries containing "wrapper" code that has to be compiled together with the precompiled part.Removed automatic '--relax' for compiles on Atmega2560 MCUFixed skethbook+bootloader hex merger when the bootloader is in
.bin format.Fixed relative path errors when traversing different partitions (in particular on macosx for larger sketches)Added #define variable during lib discovery phaseImproved progress report from compile[core]Updated AVR core to 1.8.3:Wire class now supports timeoutUpgraded avr-gcc to 7.3.0-atmel3.6.1-arduino7ARDUINO 1.8.12
2020.02.13[ide]Reverted the bundled Java JRE to the latest freely redistributable Oracle JRE for all platforms except:macOS (due to notarization)aarch64 (due to missing graphical support)Switching to AdoptJDK demonstrated to be more challenging than expected, bringing in some corner case bugs that were not being caught by our testing
procedures: guarantee the best user experience we decided to revert to the latest working JREuntil all the issues are resolved.Improved visualization of compiler errors (@matthijskooijman)Fixed: "Comparison method violates its original contract" error in library managerSerial monitor is no more cleared during upload, so it can be copy/pasted
(@PaulStoffregen)[arduino-builder]FIX REGRESSION: library priority selection for bundled libraries (like SD or Ethernet) FIX REGRESSION: use AVR core installed via Board Manager (when present) instead of the one bundled in the IDEImproved precompiled libraries handlingFIX: paths with spaces are now correctly printed in compiler
outputARDUINO 1.8.11 2020.01.27[ide]FIX: Serial plotter do not hang if empty lines are receivedA lot of accessibility improvements in Libs and Board managers GUI for screen readers (thanks @joew46167)Sligtly improved support for themes in Libs and Board managers (thanks @matthijskooijman)Serial plotter: added serial input text field (thanks
@cactorium)MacOSX: support for notarization (thanks @PaulStoffregen for the support)Fix wrong bracket match rectangle on auto format (thanks @magedrifaat)Pluggable discoveries: runtime.platform.path and runtime.hardware.path are now available as variables (thanks @PaulStoffregen)[arduino-builder]FIX: missed library-detection cache
(forcing lib detection on each build)FIX: Windows: the build folder may reside on a different partitionFIX: Segfault in arduino-builder with -dump-prefs if no sketch is specifiedFIX: Allow loading of global user-define platform.txtFIX: Fixed nil pointer exception on some rare case with multiple libsFIX: Print "multiple libraries found for header.h"
message only when it really happensFIX: Fixed library priorities on some rare circumstances[hardware]updated AVR core to 1.8.2[wifi-updater]ARDUINO 1.8.10 2019.09.13[ide]Live font resize now applies to console and serial monitor too @Pieter12345Make it easier to build the IDE from sources on Windows @Pieter12345Smaller Preference
window, suitable for supersmall screensCache downloaded json and only retrieve them when changed @mattiabertorelloSuspend serial monitor if the board gets disconnected abruptlyLibraries can now declare dependencies that will be proposed during lib installationSerialPlotter: allow adding labels @chromhelmSupport tar.xz archives for Board
Manager @vowstarSerialMonitor: add command history (via arrow keys) @Pieter12345Accessibility: add checkbox to enable accessibility in Preferences @joew46167. This mode enables a plethora of improvements on voiceover and keyboard navigation. eg. links become accessible buttons, lists are scrollable via arrow keys without much pain[wifi-
updater][hardware]Bundle new avr toolchain (fixes lto issue with 5.4.0 and 64bit compatibility with upcoming macOS Catalina)[arduino-buidler]update to 1.4.7 (based on the new arduino-cli)ARDUINO 1.8.9 2019.03.15[ide]New target! ARM64 boards are now fully supported (Nvidia Jetson and RaspberryPi3 with 64bit operating system)Fix UTF8
related bugs on WindowsFixed: can now compile again using OpenJDK (removed dependency on JavaFx)Fixed: only use touchBar on OSX 10.12 or higherPluggableDiscovery: easily add your own discoverer and make it available in Boards/Port menu (thanks @PaulStoffregen for the idea and the initial implementation)LibManager: reduce slowdown
when searchingFixed: ensure a sketch line is visible when it contains an error and must be highlightedLinux: the installer now tries to install a symlink is /usr/local/bin (this doesn't affect usual non-systemwide installation) Thanks @2E0PGS[wifi-updater]Fix firmware upgrade not being available for 3rd party WINC boards[arduino-builder]Use
lexicographic distance as last chance to spot the right library (if every other tecnique failed)Fix some prototypes being inserted in the middle of a functionARDUINO 1.8.8 2018.12.07[ide]Fixed: command line parsing of version parameters when installing cores/librariesPlatform indexes are now downloaded using httpsFixed: on some newer linux
distrubitions, NPE when loading GTK look-and-feel without libgtk2 installedMacOSX: added touch bar supportMacOSX: do not exit app after closing last windowFixed: serial monitor timestamps not always printed. Thanks @nitram509Kill active programmer if still alive after closing last IDE windowFixed: "Export compiled binary" now works also with
unsaved sketchesImproved automatic port re-selection after uploadAdded scroller to "INCOMPATIBLE" examples menu[wifi-firmware]Added firmware upgrade for NINA-based boardsARDUINO 1.8.7 2018.09.11(bugfix service release)[ide]Fixed: IDE doesn't start if library_index.json is corruped.Fixed: IDE doesn't start if a library with invalid version
is found.Fixed: Rare build errors in very complex sketches (like Marlin firmware).Fixed: Better dialog explaining that MacOSX 10.8 is now required. Thanks @PaulStoffregenFixed: Slow "File" and "Tools" menus in MacOSXFixed: Weird Board Manager behaviour if AVR core is downgraded to a version earlier than 1.6.22Improved first-use usability if
the user don't select the serial port. Thanks @PaulStoffregenCustom "Tools" menu now keeps the order as defined in boards.txt.[core]AVR: Fixed EEPROM warnings. Thanks @bxparksAVR: Treat "narrowing conversion" as warning, not error. This will make some libraries to compile again. Thanks @PaulStoffregenARDUINO 1.8.6
2018.08.23[ide]Updated arduino-builder to 1.4.0:parallel build (improve compile speed)better reuse of already compiled filesWindows: fixed build for sketch stored under OneDrive cloud foldersAdded translations for library types. Thanks @ReAlUAAdded scroller in programmers menu.Added "accelerator" in scrollable menu, press "a" key while
scrollingKeep "Board Manager" menu item always as first in the menu"Manage Libraries" is now also in the "Tools" menu. Thanks @feiknameCompiler errors now have both column and lineSerial monitor how has "Show Timestamp" option. Thanks @MichalSyImproved DPI detection for hi-resolution montior on LinuxAdded copy/paste contextual
menu in Find/Replace text fieldsFixed long IDE startup time when some virtual/not-connected network interfaces are presentFixed inconsistent line-endings when saving files on Windowsmdns: Make board disappear via TTL instead of reachability. Thanks @kurtgoLibrary Manager: display install/remove buttons with mouse on-hover instead of mouse
clickBoard Manager: fixed some rare case with corrupted index downloadWindows: removed some debug output when running the IDE from command lineSupport selectable user-defined themes. User's theme are zip files inside the sketchbook "theme" folderLibrary Manager: library search now search also for headers (.h) declared by the library in
library.propertiesAdded "--version" to the CLI. Thanks @yohsukeFixed Serial Plotter/Monitor permanently disabled if Plotter fails to openFixed window title that sometimes is not updated on "Save as...". Thanks @JxJxJxMxNxSFixed keybindings for Increase/Decrease font sizeImproved accessibility in the Preferences dialog. Thanks
@coreyknapp[core]AVR: Set Optiboot as default bootloader for Nano. This is the bootloader installed on new Arduino Nano boards. If you have a Nano with the old bootloader you must select Tools > Processor > ATmega328P (Old Bootloader) to upload.AVR core has been moved to its own repository: toolchain to gcc 5.4.0[wifi101-
firmware]SECURITY FIX: Released fix for WPA2 KRACK vulnerability (latest WINC1500 firmware 19.5.2).ARDUINO 1.8.5 2017.09.29(bugfix service release)[ide]Added workaround for menu visibility bug in MacOSX 10.13 beta. Thanks @puybaretFixed bug for negative-font-size.New/Rename tabs now allows names starting with a number.ARDUINO
1.8.4 2017.08.23[ide]IDE Toolbar can now go full width on 4K display.Environment variable LIBRARY_INDEX_URL is now correctly parsed (LIBRARY_INDEX_URL_GZ can also be optinally specified). Thanks @xardaelAdded per-board generic option in config file boards.txt for disabling control of dtr+rts. Thanks @d-a-vCLI: fixed missing
"runtime.platform.path" when running without the --board optionLinux: fixed desktop icon not launching the IDE in certain Window Managers (thanks @herrold)Allow setting low values for console size (correctly applies console.lines=x in preferences.txt)Make Preference window fit 600px height displaysFix error when renaming an unsaved, newly
added file. Thanks @matthijskooijmanAdded the ability to increase/decrease font size via keyboard + mouse shortcuts (Ctrl + MouseWheel)Sketch names starting with a digit are now allowedSerial monitor is opened on a default position if the multi-monitor setup change (this should fix cases when the serial monitor is opened outside of the visibile
space)[libraries]Fixed wrong folder name for "Adafruit Circuit Playground" library, now it can be updated cleanly.Fixed wrong include for "RobotIRRemote", now it should not interfere anymore with the real IRRemote.ARDUINO 1.8.3 2017.05.31(bugfix service release)[ide]Fixed RESOURCE_NAME in install.sh scriptSerial Monitor: added "Clear
output" button. Thanks @MichaelSyadded options in preferences.txt for: custom title, comment/uncomment shortcut, always show file extensions. Thanks @MichaelSyadded options in preferences.txt to enable/disable typing assistance (auto close brace, auto indent)Enabling "Use external editor" option now requires to save all sketches.[core]AVR:
added availableForWrite() method in generic Stream class. Thanks @eric-wieserAVR: moved flush() method up in the class hierarchy from Stream to Print class. Thanks @matthijskooijman[other]Update Wifi101 Firmware Updater pluginARDUINO 1.8.2 2017.03.22[ide]Fix command line: works again with relative paths (regression)Fix command line:
"--save-prefs" works again (regression)AVR toolchain has been updated with a tentative fix for the ld-returned-5-exit-status bugUpdate arduino-builder to 1.3.25avoid name clashing for librariescache core archives to speedup compilation consistentlyAllow BoardManager to fetch FreeBSD tools (thanks @kevans91)Serial monitor: the input string box is
automatically focused when window is selectedSerial monitor: now can not be opened during uploadSerial monitor: now properly decodes UTF8 characters (thanks @aknrdureegaesr)Serial monitor: added 500k, 1M and 2M baudrates (thanks @dsstutts)Updated RSyntaxTextArea to 2.6.1 (textarea component)Updated jmdsn (mDNS discovery)Allow
plugins to attach a listener to compile progress (thanks @tomneutens)[core]Add Atmel-ICE and JTAGICE3 programmers for AVR chips (thanks @matthijskooijman)AVR: Set unused bits of extended fuse to 1, should remove some avrdude warning during burn bootloader (thanks @descampsa)AVR: USB: send ZLP when needed (allows full 64 bytes
packets)AVR: USB: use IAD descriptors instead than Generic (thanks @cuitoldfish)[other]ARDUINO 1.8.1 - 2017.01.09[ide]Fixed font rendering not anti-aliased on Windows (regression)Increased number of colors on serial plotter to 8, thanks @cousteaulecommandant[libraries]Fixed regression in SD library. Thanks @greimanARDUINO 1.8.0 -
2016.12.20[ide]Linux: running in command line mode doesn't require an X11 display anymore"Save as" now clears the "modified" statusbuilder: Paths with strange UTF8 chars are now correctly handledbuilder: .hpp and .hh file extensions are now considered valid sketch extensionbuilder: core.a is not rebuild if not needed (improve build time in
particular for big projects)Fixed swapped actions "Copy for Forum" and "Copy as HTML"Linux/osx: If an editor tab is a symbolic link it is no more replaced with a real file when saving (see #5478)Increased the upload timeout to 5 minutes (it was 2 min, but it may be not sufficient when uploading via UART a big sketch)[core]Added Arduino.org
boardsAdded Adafruit Circuit Playground boardAdded "-g" option to linker to keep debug information in the .elf file (see #5539)avrdude: Added fake configuration for EFUSE on atmega8 part. This solves a long standing issue with "Burn bootloader". Thanks @rigelinorion, @awatterottARDUINO 1.6.13 - 2016.11.22[ide]Improved robustness of Serial
Plotter while dealing with malformed or partial data. Thanks @xloem.Fixed regression on command line upload.Bugifx installing libraries from command line: the IDE tries to update the libraries index but it didn't use it straight away (this caused issues mainly on CI environments)Libraries and Boards Managers: if a download error happens (CRC
error) the IDE tries to download the file again without the need to remove the corrupted file manually.Improved serial plotter with horizontal axis and grid. Thanks @duff2013Windows: Improved DPI detectionFixed a bunch of small bugs in the editor.[core]avr: set default values for "upload.verify" and "program.verify" (allows compatibility with older
IDE). Thanks @per1234ARDUINO 1.6.12 - 2016.09.21[ide]Refactor Editor code, solving some long standing bugs and cleaning up everything. Thanks @matthijskooijmanAdd initial Cloud API integration. When plugging a new board, if an additional core is needed, the user will be prompted to install itImproved examples menu organization. Thanks
@PaulStoffregenRecompiled arduino-builder with Go 1.7.1 to avoid crash on OSX Sierra[core]avr: Add support for internal 2.56V and 2.56V ext. cap analog references on the ATtiny25/45/85. Thanks @mischnicAdded String::toDouble(). Thanks @Ivan-PerezUpdate to avrdude 6.3 with patch to ignore unused fuses (burn bootloader works also wih
outdated cores)ARDUINO 1.6.11 - 2016.08.17[ide]ARDUINO 1.6.10 - 2016.07.26[ide][core]avr: toolchain updated to latest Atmel release 3.5.3 . The updated tools are now binutils-2.26, gcc-4.9.2, avr-libc-2.0.0, gdb-7.8avr: avrdude updated to version 6.3fixed a small bug that caused a compile error on some 3rd party derivatives boards based on
Leonardosam: added snprintf_P to avr/pgmspace.h stub. Thanks @jantjeavr: Fix TX buffer being overwritten by multiple Wire.write(...) calls in slave mode. Thanks @kellerkindtAnother small fix to String iterators. Thanks @Ivan-Perez @Chris--AAdded getTimeout() method to Stream. Thanks @mattb5906avr: Fixed USB wakeup event handling, this
solves some rare lockups of serial port on some linux distributionsSpeed and size improvement on Print::printFloat(..). Thanks @bblanchonavr: Added Serial.availableForWrite() for USB-CDC serial portsavr: Added PIN_* defines to all board variantsARDUINO 1.6.9 2016.05.10[ide]Catch and report errors during parsing contributed index filesFixed IDE
version color on about dialog box. Thanks @ivanebernalThe "always-on-top update notification" popup is now less intrusive.Fixed untraslated string during IDE startup splash window. Thanks @ivanebernalNew arduino-builder: better core library detection, faster operations when recompiling and more. See multitab error reportingRework serial port
discovery to be faster, less cpu intensive and FTDI friendlyAvoid launching multiple concurrent compile and upload operationUse hi-res icons for Serial monitor and plotterMake and links clickable from the editor windowCut/Copy actions are disable when there is no text selected. Thanks @avargas-nearsoftAdded more OSX native (emacs-like)
keybindings. Thanks @nopdotcomFixed Ctrl+Del: now deletes the word behind the cursor instead of the entire line. Thanks @avargas-nearsoftFixed "Verify code after upload" option in preferences. Thanks @gh-megabit[core]String class now supports iterators. Thanks @Chris--Asam: Allow 3rd party boards that depend on SAM core to use their own
USB vid/pid and manufacturer/product strings. Thanks @philmanofsky.avr: Check at runtime if 32u4 boards are shipped with new bootloader; if so write bootloader magic value in an unproblematic RAM locationavr, sam: Added "reciper.ar.pattern" to plaform.txt to allow compatibility with older version of Arduino IDE. Thanks
@per1234[libraries]Bridge / bridge.py: added support for SSL sockets (Yun firmware >=1.6.2 is needed).ARDUINO 1.6.8 - 2016.03.09[ide]Added support for GUI interface scaling for UHD monitors.Thanks to all contributors and testers: //github.com/arduino/Arduino/pull/2776Editor position is saved when closing with Alt+F4 or clicking on the "X"
button. Thanks @willie68Fixed a NullPointerException when dealing with some rare combination of package_*.json filesFixed incorrect key bindings handling for changing tab. Thanks @matthijskooijmanMacOSX: Fixed handling of add indent/remove indent shortcuts (CMD+[and CMD+])Fixed incorrect update of available libraries in Library
Manager. Thanks @vicnevicneSerial plotter now correctly resize graphs with negative values. Thanks @vicnevicneImproved sketch preprocessor when handling C++11 keywords. Thanks @matthijskooijman @igrrUpdates to keyword pigmentation to match CreateFixed display of LITERAL1 keywordsAdded target board information when "Error
compiling" message is displayed. Thanks @PaulStoffregen[core]avr: fixed USB_SendControl(...) for buffer with len > 64. Thanks @NicoHoodavr: fixed yield() usage inside delay(). Thanks @vbextremesam: fixed USB descriptor lengthavr: reduced boiler-plate for ISR declarations. Thanks @paulo-racaavr: added Timer3 support for atmega1284/1284p.
Thanks @feilipusam: added USB CDC send break support and readBreak API[libraries]Added support for Arduino/Genuino 101 to: TFT, Ethernet, SDSD: the library is now board-agnostic. Thanks @feilipuARDUINO 1.6.7 - 2015.12.17[ide]New arduino-builder: faster, better prototype generation and library discovery logic, and more. See plotter now
allows to plot multiple values at once. Thanks @henningpohlNew translationsAdded support to file:// protocol for boards manager URLsPortable sketchbook folder can now be any arbitrary locationFixed a bug that caused Boards Manager to install the same tool twice or more, thus wasting disk space, if other cores where using tools delivered by
arduinoFixed various Boards manager glitches, in particular when removing a boardWindows: Arduino IDE icon is now in high definition, bye bye that blurry iconRemote upload without SSH is now supported (allows OTA upload for many 3rd party boards). Thanks @me-no-dev.[libraries]SPI: Added SPI.transfer16(...) function to SAM core.Ethernet,
WiFi, SoftwareSerial: Fixed flush() behaviour: the flush function is no more dropping the receive buffer, as per 1.0 API specification. Thanks @drmpf[core]Fixed wrong timings for HardwareSerial::flush() in SAM core. Thanks @borisffStream: Optional lookahead and skipChar options added to parseInt and parseFloat, parseFloat also support parsing
floats without a leading 0, Stream::find(char) added to SAM. Thanks @Chris--A.Leonardo/Micro (and other atmega32u4 based boards) now have SerialUSB alias for SerialLeonardo now has LED_BUILTIN_RX/TX definitions in variant file. Thanks @NicoHoodARDUINO 1.6.6 - 2015.11.03[ide]Switched to Java 8, which is now both bundled and needed
for compiling the IDEAdded link to unofficial boards support list in preferencesLimit of possible new sketches in a day raised to 676. Thanks @Chris--AWindows: fixed a bug when running username had non ASCII charsLots of fixesArduinoISP examples moved under 11.ArduinoISP folderMacOSX: binary is now able to tell the IDE its current working
directory. Eases life to those using the IDE from the CLICLI: if specified board doesn't exist, fail instead of using a default oneWindows: IDE stores its stuff under %LOCALAPPDATA% instead of %APPDATA% (which was a roaming folder). %APPDATA% content is automatically migrated to new locationAdded 74880 baud rate to serial monitor. Thanks
@Chris--AWhen changing tab, file name is shown in window titleIDE groups compiled files into "libraries", "core" and "sketch" foldersAdded Serial Plotter. Thanks @henningpohlEditor: triple click selects whole lineWindows: IDE startup feels fasterExamples of libraries with a .development flag file in their root folder won't get marked as read-only.
See #2619Update notifications: when updates are available from either Library or Boards Managers, a popup will be shownIntroducing arduino-builder, a command line tool that properly calls gcc. Fixes lots of issues about buggy preprocessing.Each open sketch gets built in its own folder, allowing the IDE to recycle previous compilations and being
much faster"Go to line": CTRL+L or Edit > Go to line... will open a dialog where you can type the line number where you want the editor to jump toAdded boards.local.txt support: like platform.local.txt, allows to enrich a boards.txt definition without modifying the original file. Thanks @WackerbarthLibrary to library dependencies: when your sketch
imports a library, and that library uses another, the IDE will find out without you having to add a useless #include to your sketchAllow per-platform keywords.txt. Thanks @PaulStoffregenAllow spaces in IDE install path on Linux. Thanks @matthijskooijmanFixed a lot of documentation. Thanks to @per1234 @q2dgFixed parsing for SystemProfiler
when scanning for available serial ports on Macosx El Capitan[libraries]Bridge: YunClient.connected() returns true if there are bytes available for read. Thanks @RobAtticusBridge: YunClient.stop() now empties all buffers. Thanks @RobAtticusEthernet: Added missing keywords. Thanks @per1234Ethernet: Fixed DNSClient.inet_aton() function.
Thanks @jhorckEthernet: Expose optional DHCP timout parameters to Ethernet:begin(). Thanks @chaveiroWire: Implemented Wire.end().Wire: Fixed bug with repeated START for AVR. Thanks Nate WilliamsWire: Resolved timing issues on AVR. See #2173 #1477.Wire: Fixed bug in repeated start in slave mode on SAM. See #3908SD: Change "char "
to "const char " on methods accepting a string as a parameter. Thanks @Ivan-PerezScheduler: Added support for cortex-M0. Thanks @PeterVH[core]Added new PluggableUSB and HID support. This change introduces a more powerful API for libraries that wants use advanced features of USB-Device. Thanks to @obra @NicoHood @matthijskooijman
@nospam2000 @weizenspreuAVR: fixed wrong turnOffPWM() for TIMER0B. Thanks @gonzovelikiAVR: added support for AtmegaXXu2. Thanks @NicoHoodAVR: a bunch of USB Core bug fix. Thanks @NicoHoodAVR: Expose serial setting from USB-CDC serial port. Thanks @matthijskooijmanArduinoISP is now compatible with every architecture (not
only AVR) and is much more stable. Thanks @PeterVHPrint not aborting on write() failure. Thanks @stickbreakerAVR: Faster interrupts patch. Thanks @xxxajkAVR: Fixed pulseInLong wrong measurements.AVR: Removed a bunch of compiler warnings. Thanks @NicoHoodImproved variants files for Boards Gemma and EthernetARDUINO 1.6.5-r5 -
2015.08.28[ide]Fixed a but preventing upload on boards that do not supply VID/PID. See #3738ARDUINO 1.6.5-r4 - 2015.08.27[ide]Added support to Genuino boardsARDUINO 1.6.5-r3 - 2015.07.20[ide]Fixed Mac signature for OSX >=10.10.4ARDUINO 1.6.5-r2 - 2015.06.17[ide]Windows: fixed a problem that prevented opening the IDE when double
clicking a .ino fileARDUINO 1.6.5 - 2015.06.15[ide]File, Sketch and Tools menu items are properly handled when switching between different windowsSketch > Export compiled Binary: saves the compiled binary (hex, bin) into the sketch folder. Thanks @championswimmerNew editor, based on RSyntaxTextArea. Thanks @ricardojlrufinoNew
keywords. Thanks @Chris--AEasier "additional boards manager url" field: a wide text area opens by clicking the added button (right side of the text field)Rewritten code of Preferences window: its content is now correctly drawn on every OSFixed a bug that made the IDE notify users of invalid libraries too many times. Thanks @Chris--ARemoved JNA.
Less native stuff and less chances of incurring into an UnsatisfiedLinkErrorMany new and old issues closed. Thanks to many, and @Chris--A in particularFaster libraries list updateSerial monitor stays opened during upload, disabled. Thanks @avishorp and @WackerbarthCLI: --get-pref can now be called without a pref name. If pref name is missing,
IDE dumps all preferencesHolding SHIFT when hovering the save icon will trigger a "Save As"Removed proxy settings from File > Preferences: IDE will use system settingsLinux versions bundle the latest JVM, 1.8.0_45Local docs: if your sketch has a "docs" folder, you can add local links to it. For example file://./docs/index.html will use your browser
to open file index.html from the "docs" folder of your sketchWhen using "external editor" mode, sketch code is updated when the IDE gets focusedAdded keyboard shortcuts to IDE menus: ALT+F for File, ALT+E for Edit and so onAdded support for Dangerous Prototypes Bus Pirate as ISPAdded "Close" button to Boards/Libs Managers, in order to help
linux people with weird Window ManagersAdded File > Open Recent menu, showing the last 5 opened sketchesWindows: added Arduino Zero driversTons of minor fixes[libraries]LiquidCrystal fixes. Thanks @newbie15Added SPI Transactions to TFT libStepper: support for 5-phase/5-wires motors. Thanks @rdodesignsStepper: increased precision in
timing calculations. Thanks @ekozlenkoFirmata and Temboo: dropped our vesions, tagged released are downloaded from their respective git repos[core]AVR: delayMicroseconds(..) doesn't hang if called with 0. Thanks @cano64AVR: delayMicroseconds(..), added support for 1Mhz, 12Mhz and 24Mhz. Thanks @cano64AVR: added missing case in
detachInterrupt(). Thanks @leres, @vicatcuSAM: added watchdog routine for Due. Thanks @bobcAVR+SAM: reworked pulseIn() function to become GCC agnosticAVR+SAM: added pulseInLong() (based on micros()) to help getting good results in interrupt-prone environmentsAVR: fixed regression in HardwareSerial.flush(). Thanks
@chromhelmARDUINO 1.6.4 - 2015.05.06[ide]Added support to Arduino GemmaRestored previous "error" colour in black lower part of the IDERelaxed IDE checks against libraries versionsFixed colour of 'char' keyword. Thanks @mixaniaTools submenu show selected subentry. Thanks @PaulStoffregenBoards and Libraries Managers command line
(with contribs from @Lauszus): --install-board "arduino:Arduino SAM Boards (32-bits ARM Cortex-M3):1.6.3" and --install-library "Bridge:1.0.1". Version can be omitted: if omitted, latest version available is automatically pickedWarning levels can be set in File > PreferencesWhen compilation fails, editor highlights the reported rowWindows and
MacOSX: updated bunbled JVM to 1.8.0_45 (latest available atm)Disabled "uncertified" warning (requires avr core >= 1.6.6, sam core >= 1.6.4)Additional boards/cores can be listed specifying comma separated list of package_SOMETHING_index.json url in File > PreferencesMacOSX: fixed wrong sketch opened when double clicking on a .ino
fileBoards and Libraries main files are downloaded directly (no proxy php script), fixing some glitches when using a proxy serverAdded notification if the bundled AVR core is newer than the one installed through Boards Manager (useful when upgrading the IDE)Fixed error when in Serial and Network discovery that was leaving the Ports menu
greyed outIDE now deletes temporary files on exit. When using the CLI, --preserve-temp-files arguments skips deletion[libraries]Bridge fixesSD: allow multiple SD instances (@squelched)SD: fix examples with Ethernet shieldARDUINO 1.6.3 - 2015.04.02[ide]Fixed an ArchiveExtractor issue when dealing with native stuff, affecting some
linuxesMacOSX: MACOSX_BUNDLED_JVM property is automatically set to JAVA_HOME if not providedNew splashscreen imageNew editor themeMacOSX and Windows: https links were not working, fixed[core]Upgraded toolchains: statically compiled, ship with Link Time Optimization plugin (disabled by default)Restored backward compatibility by
providing a platform.txt in-memory rewrite: old values (eg: compiler.path) are converted to new valuesplatform.txt rewriting can be disabled if target platform has a "rewriting=disabled" key/value pair setAVR bundle files (core files and toolchain) is again inside IDE "hardware" folder, instead of unpacked into user's home folder at first IDE
startupAdvise of duplicate libraries after compiling. Thanks @PaulStoffregenARDUINO 1.6.2 - 2015.03.28[ide]Boards and libraries managers, for one click install of additional boards and librariesIn platform.txt, pre and post build hooks can now be specified. Example: recipe.hooks.prebuild.0.pattern=echo "Hello {build.source.path}". Thanks
@WackerbarthWindows and MacOSX JVM Xmx halved to 512MIntroduced starting splashscreen with progress status: will be used for notifying user of long running startup tasksAvailable ports list is now generated in background: hence "tools" menu is much fasterMacOSX: appbundler merged our contribution, switching to upstream version
core]Stream: fixed bug in findUntil routine #2591 @Xuth[libraries]EEPROM: Replaced existing library with more complete implementation @Chris--ASD: fixed endless timeout on 32bit core (Due) @TimmmmEthernet: fixed headers include pathSPI: do not influence state of SS pin if it's already been set as output by user sketch @vicatcuBridge:
discard message if received length is bigger than rx buffer @arve0ARDUINO 1.6.1 - 2015.03.10[ide]Improved Yun detection for upload via network (Ron Guest)In platforms.txt "objcopy" recipe is no more tied to the "hex" format (Arnav Gupta)/dev/cu.* serial ports are now filtered from the port list on MacOSXPorts in ports list are now grouped by
typeUpgraded avr-gcc toolchains to 3.4.5Fixed: wrong parsing of boards.txt when using submenu and boards id with underscoresWindows: updated bundled JVM to latest 8u31Windows: JVM memory settings can be customized editing arduino.l4j.ini and arduino_debug.l4j.ini files"new" button now behaves as "File -> New..." menu: a new window with
an empty sketch is createdWindows: opening a lonely sketch caused the IDE to crash. Fixed with a bounty on bountysource @gerito1Bridge: fixed bug with multiple and failing YunClient.connect calls @snglTranslations updatedImproved SoftSerial library and optimized its memory usage @matthijskooijmanFixed: status board was not changing when

using custom menu @PaulStoffregenFixed: better error message when using a busy serial deviceFixed: missing bool operator on EthernetClientFixed: missing bool operator on SoftSerialMacOSX: back to native file dialogs and buttons, when using experimental versionMacOSX: command line interface now works also with java-latest versionImproved library name matching @PaulStoffregen. Avoids conflicts with libraries that bundle files equally namedARDUINO 1.6.0 - 2015.02.09[ide]Updated help with the latest docsUpdated translationsARDUINO 1.6.0rc3 - 2015.02.03[ide]ARDUINO 1.6.0rc2 - 2015.01.20[ide]Reenabled speed of 38400 on serial monitorImproved Find/Replace dialog layout (Eberhard Fahle)Fixed missing .dll error on some Windows box[core]Arduino "boolean" type is now mapped to "bool" instead of "uint8_t" (Christopher Andrews)sam: HardwareSerial now has buffered transmission (Collin Kidder)sam: HardwareSerial fixed modes (parity, data bits, stop bits) (bluesign2k)avr: Fixed regression in pulseIn() function accuracy[libraries]The following changes are included also in the Arduino IDE 1.0.7:[ide]Mitigated Serial Monitor resource exhaustion when the connected device sends a lot of data (Paul Stoffregen)ARDUINO 1.6.0rc1IDE internals have been refactored and sorted out. (Claudio Indellicati) . The builder and the GUI now have clearly separated source code making it easier for IDE developers to maintain and propose patch. From a user point of view nothing changes and the IDE should behave exactly as before.[ide]Autosave on sketch Upload/Verify (Fulvio Ieva)Sketch build process: fixed full rebuild on windows even if not neededSketch build process: core.a rebuild only if needed (Matthijs Kooijman)Updated AStyle formatter to v2.04: avrdude verbose upload (Matthijs Kooijman)(Mac OSX) Add Exported UTI for ino files, allows quick look to view the content of the file and external editors to syntax highlight as C++ (Matt Lloyd)Updated libastyle to 2.05[core]sam: added -MMD flag to let gcc produce dependency files (full rebuild on Arduino Due is now triggered only if needed)[libraries]LiquidCrystal: added setRowOffsets function to support different LCD hardware configurations (Mark Sproul)LiquidCrystal: various improvements and optimizations (Matthijs Kooijman)Fixed PROGMEM error in Robot_Control/examples/explore/R06_Wheel_CalibrationSD: Fixed SPI transaction mismatch (Paul Stoffregen)The following changes are included also in the Arduino IDE 1.0.7:[libraries]EthernetClien: use IANA recommended ephemeral port range, 49152-65535 (Jack Christensen, cifer-lee)[core]ARDUINO 1.5.8 BETA - 2014.10.01[ide]Find/Replace over multiple tabs (Erwin Ried, Fulvio Ieva)Fix wrong "Replace" behavior with empty "Find" field (Fulvio Ieva)A lot of command-line improvements (Matthijs Kooijman)Cursor is not moved when invoking autoformat[libraries][core]sam: "new" and "delete" do not bring in 50Kb of stdclib anymoreCorrect implementation of gcc specific internal functions (_cxa_guard*) (Alexey Guseynov)Fix for upload problems on Leonardo (and derivative boards based on mega32u4)Small improvements on String class (Matthijs Kooijman)Fixed size of available flash for mega2560 boards (Leonardo Milani, Arnav Gupta)sam: removed useless empty digitalPinToTimer(..) macro stub (Bill Perry)The following changes are included also in the Arduino IDE 1.0.6:[core]avr: Added replacement stub for cstdlib atexit() funciton (Christopher Andrews)ARDUINO 1.5.7 BETA - 2014.07.07[core]Upgraded AVR toolchain: gcc 4.8.1, avr-libc 1.8.0Upgraded ARM toolchain: gcc 4.8.3-2014q1Upgraded avrdude to version 6.0.1ARM gcc doesn't require ia32-libs anymore on 64 bits linux systemsavr: fixed typo in SerialEvent3 handling (Matthijs Kooijman)avr: HardwareSerial support for different size of TX and RX buffer sizes (Jan Baeyens)avr: HardwareSerial support for buffer sizes bigger than 256 bytes (Jan Baeyens)sam: Added configuration (parity, data bits, stop bits) to Serial1/2/3 of Arduino Due (bluesign2k)Removed a lot of compiler warnings from Arduino coreavr: Fix EXTERNAL_NUM_INTERRUPTS for atmega128rfa1 and atmega256rfr2 (Matthijs Kooijman)sam: Fix to Wire::endTransmisson() return value (bluesign2k)sam: Fix to Wire usage of TWI status register (bluesign2k)avr: Fixed PROGMEM statements to be compatible with newer avr gcc (Scott Howard)[ide]Moved to appbundler for building releases for MacOSX. (Haavar Valeur). This should remove dependency from Java 1.6 on recent MacOSX.Added support for '-' and '.' in filenames (Georg von Zengen)(re)Added 'arduino_debug.exe' in Windows build for debugging purposesMagic baudrate is no longer removed (it was a workaround for RXTX) (for more info see github issues: #1203 and #995)Allow overriding platform.txt using platform.local.txt (Matthijs Kooijman)Explicitly define compiler.path in avr/platform.txt (Matthijs Kooijman)Make the low available memory message a warning (Matt Robinson)Proceed with upload even if port can't be found (David Mellis)Added support for ArduinoISPWindows: added board detection on serial port menu[libraries]Updated SpaceBrew libraryFixed HttpClient::running() functionFixed HttpClient::ready() function (Manuel Rabade)Added HttpClient::noCheckSSL() methodImproved speed of YunSerialTerminalFixed CRC of shutdown command on YunSerialTerminal exampleUpdates/Fix to various examplesAdded Wire.setClock(..) method (Kristian Sloth Lauszus)The following changes are included also in the Arduino IDE 1.0.6:[core]avr: Improved USB-CDC write speed (Justin Rajewski)avr: Improved USB-CDC read code (Paul Brook)avr: Fixed race condition in USB-CDC transmit (Paul Brook)Fixed wrong NULL pointer handling in Stream class (Amulya Kumar Sahoo)Added initVariant() hook to allow 3rd party variant-specific initialization[ide]Fix toolchain command line to compile assembler files (Jimmy Hedman)If two libraries have the same header file use the lib with the same folder name (Paul Stoffregen)[libraries]Robot_Control: removed duplicated SPI and Wire (Xun Yang)Robot_Control: fixed issue on motors being opposite (Xun Yang)Robot_Control: updated turning algorithm (Xun Yang)Esplora: added reading form Tinkerkit inputsSoftwareSerial: Fix idle level when initializing with inverted logic (Jens-Christian Skibakk)[firmware]Wifishield: fixed paths on firmware upgrade scriptsARDUINO 1.5.6 BETA - 2014.02.20[ide][libraries]TFT: warning messages in PImage class and strings inside examples now stored in flash to save RAM.Ethernet: added operator == for EthernetClient class (Norbert Truchsess)Robot_Control: removed embedded copies of Wire and SPI to allow their use in sketch. Old sketches using Robot_Control now must include Wire.h and SPI.h.Robot_Control: fixed issues about motor turning algorithmBridge: optimized FileIO.doBuffer() (bobh66)[core]avr: Pile of HardwareSerial improvements (Matthijs Kooijman)Remove duplicate code from HardwareSerial::begin() methodsSimplify HardwareSerial::begin() (Rob Tillaart)Use constants for register bit positions in HardwareSerialFix HardwareSerial::flush() when interrupts are kept disabled for a whileFix lockup when writing to HardwareSerial with interrupts disabledDisable the UDRE interrupt sooner in HardwareSerial (Nathan Blackwell)Put each HardwareSerial instance in its own .cpp file in order to save memory for unused UARTSIn HardwareSerial::write, bypass the queue when it's empty in order to improve throughput when using high baudrates (KurtE)sam: Fixed wrap-around bug in delay() (Mark Tillotson)sam: Fixed regression in analogRead() (fails to read multiple channels) (Mark Tillotson)sam: Fixed loops in Reset.cpp (Matthijs Kooijman)sam: Optimized delayMicroseconds() (Rob Tillaart) #1121sam: added itoa() and related functionsOptimized Print::print(String&) method, now uses internal string buffer to perform block writeImproved portability of String class (maniacbug) #695Make some operators in IPAddress const (Matthijs Kooijman)Fix for compiling assembler files with newer gccARDUINO 1.5.5-r2 BETA - 2014.01.10Signed drivers for Windows 8.1Fixed Windows drivers signature (that prevented installation on some Windows 8.x OS). Now the signature is timestamped and should not expire.ARDUINO 1.5.5 BETA - 2013.11.28NOTICE:The 1.5 library format is under heavy review on the Arduino Developers mailing list.We suggest to delay the adoption of the new format until a stable 1.5.x is released.[ide]Added missing keywords: YunServer, YunClient, powImproved folder ignore: now hidden, starting with a dot, and SCCS folders are ignoredImproved auto format: now handled by Artistic Style SSH uploader (Arduino Yun): uses file ~/.ssh/config if available. Allows public key only authentication and ssh servers listening on ports other than 22.Font anti-aliasing now ON by default (users will have to either delete or manually edit their preferences.txt file)Added warning message when opening sketches with additional files with illegal names(windows) Upgraded embedded Java runtime to version 7: much faster startupImproved files open/save dialogsAdded line numbers, can be enabled/disabled from preferences (Cayci Gorlitsky)Removed check on forbidden file in 1.5 formatted libraries (#1692)(windows) Updated msys-1.0.dll to the latest version (Windows 8.1 compatibility)Using [code] tag instead of [quote] in "Copy for forum". Removed color coding.When an hardware addon reference another core also the libraries are referenced.It's now possible to specify additional custom board parameters from command line.Sketches path can be relative (#1493)Example and Boards are now scrollable.Fixed localization issues that sometimes prevented the IDE to start.Added "Using library..." debugging message in verbose compile.Improved preprocessor (fixes #1653, #1687, #817, #1293, #1245)Fixed StringStartsWithEndsWith, KeyboardMessage, LCD CustomCharacter, Blink without delay, SD List files, Arduino Robot examplesAdded PID/VID for older Arduinos(mac) Improved board autodetection on Mac OS X[libraries]Bridge: Fixed buffer overflow in File::doBuffer() (dreggy)Bridge: Fixed timeout in Bridge::transfer()Bridge: Fixed return value of Bridge::get(..) (returning 0 with valid data len >0)Bridge: Implemented multi-byte File::read(..)sam: Fixed SPI initialization (when using extended API and multiple CS)Partially fixed the behavior of EthernetClient::flush(), now it doesn't discard incoming data BUT it doesn't wait for outgoing data to be successfully transmitted.Esplora: Added support for display GREENTAB_L[core]sam: Fixed wrong initialization for ADC timings (analogRead speed Arduino DUE improved by a factor x10) avr: Fixed wrong extended fuses on Arduino Yun when using external programmerAdded LED_BUILTIN metadata in variant filesavr: Refactored Mega* variants definitions in boards.txt (Kristian Sloth Lauszus)avr: Removed File object destructor. In order to free memory allocated by a File object the safest way is to call its close() methodAdded digitalPinToInterrupt variant macro (Paul Stoffregen)avr: added variant file for Arduino EthernetAdded SERIAL_* metadata in variants files (Paul Stoffregen)ARDUINO 1.5.4 BETA - 2013.09.10[ide]Revert to English locale if the system default is not availableAdded support for Arduino YunAutodetect of boards connected to each serial portNetwork upload and discovery for Arduino Yun[libraries]sam: fixed wrong SPI initialization (noblepepper)updated all instances of Adafruit_GFX to the latest versionadded libraries for Arduino Yun[core]avr: fixed bug introduced with recent optimizations in HardwareSerial (atmega8 cpu) (darryl)ARDUINO 1.5.3 BETA - 2013.08.30[ide][arduino core]sam: Fixed delayMicrosecond() when interrupts are disabled (Paul Stoffregen)sam: Upgraded libsam, and added missing modules (CAN, ETH, etc.) (Thibaut Viard)sam: Added compatibility for avr/pgmspace.h (Paul Stoffregen)sam: Added serialEvent*() supportsam: Fixed micros() to work with inside interrupts. (stimmer)avr: Added support for Flash strings on String class (Jantje)Added support for floating point numbers in String class (Tevin Zhang, SebiTimeWaster)sam: Fixed String buffer overflows (Paul Stoffregen)avr: Added recipe for assembly files (C. A. Church)avr: Use analogPinToChannel if it's defined (Kristian Sloth Lauszus)avr: Optimized HardwareSerial buffer (Matthijs Kooijman)removed unused flags from String (free 1 byte of SRAM)[libraries]sam: Removed CAN library, you can find an updated version here: Bugfix SPI library: begin() after end() now works (stimmer)sam: Bugfix SPI library: incorrent pin configuration in non-extended mode.Ported all libraries to new 1.5 formatUpdated Firmata to version 2.3.6 (Jeff Hoefs)[firmwares][other]Merged all improvements made in Arduino IDE 1.0.5ARDUINO 1.5.2 BETA - 2013.02.06[ide]Scrollable editor tabs (Shigeru KANEMOTO)Scrollable menusPortable Arduino (Shigeru KANEMOTO)Default sketchbook folder for linux is now "Arduino"Fixed IDE startup bug "Menu has no enabled items"Command line build.Fixed some language strings (Shigeru KANEMOTO)Fix to boards.txt: added Micro and fixed Lilypad bootloader pathRemoved check for ".h" existence in libraries.Deleting tab from IDE does not delete from temporary folderFixed NPE when unknown boards/platforms are selected in preferencesExtended command line build flagsUndo/Redo move cursor and focus to where the code has been undone/redone[arduino core]sam: attachInterrupt() now works also on pins that belongs to PORTDsam: portOutputRegister() is now writeable.sam: fixed issue on weak-symbol for some interrupt handlerssam: fixed BSoD on some Windows machine (louismdavis)sam: added CANRX1/CANTX1 pins 88/89 (same physical pin for 66/53)sam: fixed analogWrite when used in very thight write loops (V.Dorrich)sam: fixed SerialUSB.write() while sending big buffers (Bill Dreschel)sam: SerialUSB receive buffer size is now 512 (PeterVH)sam: Fixed SerialUSB data handshake when host sends a lot of data (PeterVH, stimmer)[libraries]sam: Added Servo librarysam: Added WiFi libraryFixed USBHost MouseController example[other]Merged all improvements made in Arduino IDE 1.0.3Merged all improvements made in Arduino IDE 1.0.4 (not yet released)ARDUINO 1.5.1r2 BETA - 2012.11.06Fixed wrong release file for windows.ARDUINO 1.5.1 BETA - 2012.11.05Merged changes coming from stable release of Arduino IDE 1.0.2.[arduino core]added support for yield() function in Arduino Core. This is apreliminar step for experimenting with cooperative Schedulers.[ide]Simplified boards menu selection. (new boards.txt file format withcustom sub-menus).Fixed "FAT" (multi-platform) library detection. This bug causedmany libraries to be detected as 'invalid'.[sam arduino core]attachInterrupt() and detachInterrupt() are now working.Fixed macros to query low level GPIO registers:digitalPinToBitMask(), digitalPinToTimer(),portOutputRegister, portInputRegister()Fixed analogWrite() on DAC0/1 when writing on both DACs(thanks to smay4finger)[firmwares]Added firmware for atmega16u2 on Due Board.ARDUINO 1.5 BETA - 2012.10.22ARDUINO 1.0.6 - 2014.09.16[core]avr: Improved USB-CDC write speed (Justin Rajewski)avr: Improved USB-CDC read code (Paul Brook)avr: Fixed race condition in USB-CDC transmit (Paul Brook)avr: Added replacement stub for cstdlib atexit() funciton (Christopher Andrews)Fixed wrong NULL pointer handling in Stream class (Amulya Kumar Sahoo)Backported String class from IDE 1.5.x (Matt Jenkins)Backported Print class from IDE 1.5.xBackported digitalPinToInterrupt macro from IDE 1.5.xAdded initVariant() hook to allow 3rd party variant-specific initializationAdded replacement stub for cstdlib atexit() function (Christopher Andrews)[ide]Added compatibility for IDE 1.5.x libraries layoutFix toolchain command line to compile assembler files (Jimmy Hedman)If two libraries have the same header file use the lib with the same folder name (Paul Stoffregen)[libraries]Robot_Control: removed duplicated SPI and Wire (Xun Yang)Robot_Control: fixed issue on motors being opposite (Xun Yang)Robot_Control: updated turning algorithm (Xun Yang)Esplora: added reading form Tinkerkit inputsSoftwareSerial: Fix idle level when initializing with inverted logic (Jens-Christian Skibakk)fixed a bunch of examples[firmware]Wifishield: fixed paths on firmware upgrade scriptsARDUINO 1.0.5-r2 - 2014.01.08Signed drivers for Windows 8.1Fixed Windows drivers signature (that prevented installation onsome Windows 8.x OS). Now the signature is timestamped and shouldnot expire.ARDUINO 1.0.5 - 2013.05.15[core][avr] malloc bug: backported avr-libc 1.8.0 implementation[avr] removed deprecated interrupt handlers causing compiler issueswith newer avr-gcc.[avr] added c_str() method to String[avr] Stream "_timeout" field and related methods are now protected[libraries]Upgrades to WiFi libraryFixed a bunch of examplesAdded Arduino Robot librariesAdded TFT display library[firmwares]Upgrades to WiFi firmwares[ide]Backport from 1.5: install Library from .zip file or folderUpdated windows driversAdded Windows installerARDUINO 1.0.4 - 2013.03.11[core]Fixed malloc bug (Paul Stoffregen)[libraries]Fixed memory leak when calling Ethernet.begin() multiple times.Fixed SD example listfiles.inoFixed a lot of Esplora examplesAdded GSM library[environment]Sort entries in preferences.txt (Shigeru Kanemoto)Fixed some wrong translationsFixed NPE due to permissions IO errorUpdated drivers for Windows (all-in-one, signature for Win8)[bootloaders]ARDUINO 1.0.3 - 2012.12.10[hardware]Added support for the Arduino Esplora[environment]Signed application for MacOSX 10.8[core]Fixed power-up-starts-bootloader in Leonardo (and derivative) bootloaders. (Kristian Lauszus)(Fixed digital_pin_to_timer_PGM array in Leonardo variant.Published updated Wifi firmwareUpdated source code for atmega8 bootloader[libraries]ARDUINO 1.0.2 - 2012.11.05[hardware][environment][core / libraries][build]Fix Linux arduino script when there are spaces in directory name.Pass Linux command line arguments from arduino script through to the Java code. (Allows specifying sketch to open on command line.)[examples]Included examples for the starter kit.ARDUINO 1.0.1 - 2012.05.21[environment][core / libraries][examples]ARDUINO 1.0 - 2011.11.30[environment]The file extension for sketches has changed from .pde to .ino, to avoid conflicts with the Processing software ("ino" are the last three letters in "Arduino").There are a new set of toolbar icons, including a checkmark icon to verify (compile) a sketch and an arrow for upload. The serial monitor icon has moved to the right of the toolbar. Also, shift-clicking the upload icon now uploads using a programmer (selected in the Tools menu). You can still enable verbose output in the preferences dialog. (Icons were designed by Nicholas Zambetti.)Theres a new color scheme and about image for the IDE (by ToDo.to.it).The name of the currently selected board and serial port are now shown at the bottom of the editor. (Code from Wiring.)A progress bar is displayed during compilation and upload. (from Wiring.)[core / libraries]Serial transmission is now asynchronous - that is, calls to Serial.print(), etc. add data to an outgoing buffer which is transmitted in the background. Also, the Serial.flush() command has been repurposed to wait for outgoing data to be transmitted, rather than dropping received incoming data.The behavior of Serial.print() on a byte has been changed to align it with the other numeric data types. In particular, it will now print the digits of its argument as separate ASCII digits (e.g. '1', '2', '3') rather than a single byte. The BYTE keyword has been removed. To send a single byte of data, use Serial.write() (which is present in Arduino 0022 as well).The Serial class (as well as other classes inheriting from Stream, like EthernetClient, SoftwareSerial, Wire and more) now contains functions for parsing incoming data, based on the TextFinder library by Michael Margolis. They include find() and findUntil() to search for data, parseInt() and parseFloat() for converting incoming characters into numeric values, and readBytes() and readBytesUntil() for reading multiple bytes into a buffer. They use a timeout that can be set with the new setTimeout().The SoftwareSerial class has been reimplemented, using the code originally written for the NewSoftSerial library by Mikal Hart. This allows for multiple simultaneous instances, although only one can receive at a time.Support has been added for printing strings stored in flash (program memory) rather than RAM. Wrap double-quoted strings in F() to indicate that they should be stored in flash, e.g. Serial.print(F("hello world")).The String class has been reimplemented as well, by Paul Stoffregen. This new version is more memory-efficient and robust. Some functions which previously returned new string instances (e.g. trim() and toUpperCase()) have been changed to instead modify strings in place.Support for DHCP and DNS has been added to the Ethernet library, thanks to integration by Adrian McEwen. Most classes in the Ethernet library have been renamed to add a "Ethernet" prefix and avoid conflicts with other networking libraries. In particular, "Client" is now "EthernetClient", "Server" is "EthernetServer", and "UDP" is "EthernetUDP". A new IPAddress class makes it easier to manipulate those values.The UDP API has been changed to be more similar to other libraries. Outgoing packets are now constructed using calls to the standard write(), print(), and println() functions bracketed by beginPacket() and endPacket(). The parsePacket() function checks for and parses an incoming packet, which can then be read using available(), read(), and peek(). The remoteIP() and remotePort() functions provide information about the packets origin. (Again, thanks to Adrian McEwen for the implementation.)The Wire library has also been modified to use the standard read() and write() functions instead of send() and receive(). You can also use print() and println() for outgoing data.The SD library now supports multiple simultaneous open files. It also provides the isDirectory(), openNextFile(), and rewindDirectory() functions for iterating through all the files in a directory. (Thanks to Limor Fried.)[boards / firmwares]Added the Arduino Mini w/ ATmega328.Added Windows drivers (.inf files) and 16U2 firmware (.hex files) for the rev. 3 boards (Uno, Mega, and Mega ADK). [internals]The WProgram.h file, which provides declarations for the Arduino API, has been renamed to Arduino.h. To create a library that will work in both Arduino 0022 and Arduino 1.0, you can use an #ifdef that checks for the ## ARDUINO constant, which was 22 and is now 100. For example:#if defined(## ARDUINO) && ## ARDUINO >= 100#include "Arduino.h"#else#include "WProgram.h"#endifThe write(), print(), and println() functions in Stream now return a size_t (instead of void). This indicates the number of bytes actually written by the function. Any classes that inherit from Stream will need to change accordingly. Additionally the write(str) function has been given a concrete implementation it calls write(buf, len) - so sub-classes don't need to (and shouldn't) implement it.There are new abstract base-classes for Client, Server, and UDP to provide portability across networking libraries.The pin definitions for the Arduino boards (i.e. the mappings from pin numbers to port register / bit pairs) is now stored in a sub-folder of a new variants/ folder in the hardware folder. The variant to use for a given board is specified by the BOARD.build.variant preference in the boards.txt file.The new, variant-specific pins_arduino.h files now provides additional macros with information about the characteristics of the board (e.g. the locations of the SPI and TWI pins, and the number of digital and analog pins).The avrdude included with the Mac and Windows versions of the Arduino software has been upgraded to avrdude 5.11 (from an Arduino-specific version of avrdude 5.4). The software now uses the "arduino" programmer type in place of "stk500v1" for uploading to most Arduino boards.ARDUINO 0023 - 2011.11.09Added .inf files for R3 boards.Updated to optiboot 4.4.Included combined firmwares for ATmega16U2.ARDUINO 0022 - 2010.12.24[core / libraries]Adding an SD card library based on sdfatlib by Bill Greiman and the MemoryCard library by Philip Lindsay (follower) for SparkFun. character manipulation macros (from Wiring): isAlphaNumeric(), isAlpha(), isAscii(), isWhitespace(), isControl(), isDigit(), isGraph(), isLowerCase(), isPrintable(), isPunct(), isSpace(), isUpperCase(), isHexadecimalDigit(), toAscii(), toLowerCase(), toLowerCase(). String.toInt() function.Refactoring core to use register-based, not CPU-based, #ifdefs. Patch by Mark Sproul. of serial baud rate calculation to match bootloader and 8U2 firmware at 57600 baud. bug in tone() function. SPI.setClockDivider() function. EEPROM library on Mega 2560. serial receive interrupt optimization. the timeout parameter of pulseIn() during measurement of the pulse, not just while waiting for it.[environment]ARDUINO 0021 - 2010.10.02Modifying VID / PID combination in 8U2 firmwares.Fixing analogWrite() bug on pins 9 and 10 (Arduino Uno).Patched RXTX to include /dev/ttyACM* on Linux.ARDUINO 0020 - 2010.09.27Added support for the Arduino Uno and Arduino Mega 2560.Including ATmega8U2 firmware used by Uno and Mega 2560.Including source code to optiboot bootloader used by the Uno.Including source code to the stk500v2 bootloader used by Mega 2560.New application icon (by ToDo).ARDUINO 0019 - 2010.09.03[core / libraries]Revised Ethernet library (by Christian Maglie). This depends on the new SPI library, so existing sketches will need: #include added to the top of their code.[environment][examples]Re-organized and numbered for easier access and better progression.ARDUINO 0018 - 2010.01.29[core / libraries]Added tone() and noTone() functions for frequency generation.Added Serial.end() command.Added precision parameter for printing of floats / doubles.Incorporated latest version of Firmata.Fixed bug w/ disabling use of the RW pin in the LiquidCrystal library. No longer disabling interrupts in delayMicroseconds().Fixed bug w/ micros() returning incorrect values from within an interrupt.Fixed bug that broke use of analog inputs 8-15 on the Mega.[environment]Synchronized with the Processing 1.0.9 code base, bringing various fixes, including to a bug causing saving to fail when closing the last sketch.Added support for third-party hardware in the SKETCHBOOK/hardware folder, mirroring the current structure of the hardware folder in Arduino.Added Ctrl-Shift-M / Command-Shift-M shortcut for serial monitor.Hold down shift when pressing the Verify / Compile or Upload toolbar buttons to generate verbose output (including command lines).Moving build (on upload) from the applet/ sub-folder of the sketch to a temporary directory (fixing problems with uploading examples from within the Mac OS X disk image or a Linux application directory).Fixed bug the prevented the inclusion of .cpp and .h (or .c and .h) files of the same name in a sketch.Improved the Mac OS X disk image (.dmg): added a shortcut to the Applications folder, a background image with arrow, and new FTDI drivers.ARDUINO 0017 - 2009.07.25[documentation / examples]Many new and revised examples from Tom Igoe.[core / libraries]Updated LiquidCrystal library by Limor Fried. See reference for details.Updated Firmata library to version 2.1 (rev. 25).Replaced the Servo library with one (MegaServo) by Michael Margolis. Supports up to 12 servos on most Arduino boards and 48 on the Mega.Improving the accuracy of the baud rate calculations for serial communication (fixing double-speed problems on 8 MHz Arduino boards). Thanks to gabebear.[environment]Synchronized with the Processing 1.0.3 code base (rev. 5503), bringing many improvements (listed below).New icons and about image by Thomas Glaser (envis precisely).Support for multiple sketch windows.The serial monitor now has its own window.Comment / Uncomment menu item (in Edit) and keyboard shortcut.Increase and Decrease Indent menu items (in Edit) and keyboard shortcuts.Support for third-party libraries in the SKETCHBOOK/libraries folder.Libraries are now compiled with the sketch, eliminating the delay when switching boards and the need to delete .o files when changing library source code.Arduino now comes as an app file (in a dmg) on the Mac.Adding the Arduino Nano w/ ATmega328 to the Tools > Board menu.ARDUINO 0016 - 2009.05.30[documentation / examples]New communication examples (w/ corresponding Processing and Max/MSP code) by Tom Igoe.[core / libraries]Adding support for the Arduino Pro and Pro Mini 3.3V / 8 MHz w/ ATmega328.Adding support for the LilyPad Arduino w/ ATmega328.Adding write(str) and write(buf, size) methods to Print, Serial, and the Ethernet library Client and Server classes. This allows for more efficient (fewer packet) Ethernet communication. (Thanks to mikalhart.)Improvements to the way the Ethernet library Client class connects and disconnects. Should reduce or eliminate failed connections and long timeouts. (Thanks to Bruce Luckcuck.)Optimizing the timer0 overflow interrupt handler (used for millis() and micros()). Thanks to westfw and mikalhart.Fixing bug that limited the bit() macro to 15 bits. Thanks to Paul Badger.Adding ## ARDUINO version constant (thanks to prodding from mikalhart).[environment]Ordering the items in the Tools > Board menu.Adding "Copy as HTML" command to the Tools menu.Eliminating (maybe) the occasional "Couldn't determine program size" errors. Thanks to the Clever Monkey.Moving selection of Linux look-and-feel into the arduino script so it can be changed by users. Thanks to Eberhard Fahle.[tools]Adding automatic dependency generation to the Makefile. (Lars Immisch)ARDUINO 0015 - 2009.03.26[core / libraries]Adding support for the Arduino Mega (ATmega1280).[environment]Reinstating use of core.a library in the build process, slightly shrinking compiled sketch sizes. (Thanks to William Westfield.)Fixing bug in copy for forum (thanks to eried).ARDUINO 0014 - 2009.03.07[core / libraries]Fixing bug that prevented multiple outgoing Client connections with the ethernet library.[environment]Clarifying ATmega168 vs. ATmega328 in the Tools > Boards menu.[tools]Updating the Mac OS X AVR tools to AVR MacPack 20081213. This includes avr-gcc 4.3.2, which should fix problems with functions called from within interrupts.ARDUINO 0013 - 2009.02.06[documentation / examples]Adding examples for Parallax Ping Sensor and Memsic 2125 accelerometer.[core / libraries]Adding support for the ATmega328. The upload speed is 57600 baud, so you may need to edit boards.txt or reburn your bootloader if you bought an ATmega328 w/ bootloader from adafruit or other supplier.Adding support for printing floats to Print class (meaning that it works in the Serial, Ethernet, and LiquidCrystal classes too). Includes two decimal places.Added word, word(), bitRead(), bitWrite(), bitSet(), bitClear(), bit(), lowByte(), and highByte(); see reference for details.Working around problem that caused PWM output on pins 5 and 6 to never go to 0 (causing, for example, an LED to continue to glow faintly).Removing cast macros, since function-style casts are a feature of C++. This should fix contributed libraries that broke in Arduino 0012.Modifying pulseIn() to wait for a transition to start timing (i.e. ignoring any pulse that had already started when the function was called).Fixing bug in random() that limited the ranges of values generated. Thanks to Mikal Hart.Modifying delay() to pause for at least the given number of milliseconds.Fixing bug in Ethernet library that interfered with use of pins 8 and 9.Originating each outgoing network connection from a different port (in the Client class of the Ethernet library). Thanks to Paul and joquer.Updating ATmega168 bootloader to work with standard distributions of avrdude (responding to signature requests made with the universal SPI command) and correctly store EEPROM data. Thanks to ladyada.[environment]Omitting unused functions from compiled sketches, reducing their size.Changing compilation process to allow for use of EEMEM directive (although not yet uploading EEPROM data).ARDUINO 0012 - 2008.09.18Added Arduino Nano to the boards menu.Arduino Pro or Pro Mini (8 MHz) to the boards menu.Added Firmata library by Hans Steiner and others. This provides a standard protocol for communicating with software on the computer.Added an Ethernet library for use with the Arduino Ethernet Shield.Added a Servo library based on the work of Jim Studt.Added a LiquidCrystal library based on the work in the playground. It supports both 4- and 8-bit modes.Improved millis(): it now overflows after 49 days instead of 9 hours, but now uses slightly more processing power.Fixed reversing direction bug in Stepper library. (Thanks to Wayne Holder.)Moved insertion of #include to after any comments and #include statements in the main sketch file. This means that an #include now works.Upgraded to newer versions of avr-gcc (4.3.0) and avr-libc (1.6). This provides support for newer Atmel chips, but may increase the size of sketches.Allowing per-board specification of the upload.using preference, allowing upload via bootloader to some boards and via a programmer to others.Added return values to some functions in the Wire library to allow for better error handling.Fixed random() to work with long values.Creation of an abstract Print base-class to allow Serial, SoftwareSerial, and LiquidCrystal to share code for print() and println().Incorporated ladyada's watchdog timer mods to the bootloader source, but only compiling them in for the Pro and Pro Mini (because they are included in the bootloader being burned on the boards by SparkFun).ARDUINO 0011 - 2008.03.28Fixed Find in Reference.Added map() function for mapping values from one range to another.Added analogReference() function.Added interrupts() and noInterrupts() functions.Added degrees() and radians() functions.Added timeout parameter (in microseconds) to pulseIn(); default is 1 second.Support for uploading sketch using a programmer.Improved detection of functions that need prototyping.Placing function prototypes after #include's and #define's.No longer moving #include statements to the top of the sketch.Can now drag .pde files onto the Arduino dock icon on Mac OS X. Thanks to Peter Sgouros.New script for downloading the reference from Tom Pollard. Thanks Tom!Miscellaneous Mac OS X and other patches from Wim Lewis. Thanks Wim!Updated Mac OS X FTDI drivers.ARDUINO 0010 - 2007.10.11Support for the LilyPad Arduino.Vista support.Mac OS X universal distribution.Printing!Copy for discourse.New Board menu replaces the Microcontroller menu.New Burn Bootloader menu offers a choice of programmers.New and improved keyboard shortcuts.Fixed some find/replace bugs.Better auto-format.Improved error messages when uploading.Support for COM10 and higher on Windows.Fixed automatic refresh of the Serial Port menu under Windows.Quaqua look-and-feel on Mac OS X.Reorganization of the Arduino application directory.ARDUINO 0009 - 2007.08.06Added support for the Arduino Diecimila.Switched to using avrdude (instead of uisp) for uploading sketches.Added the ability to burn NG and Diecimila bootlaoders (with an AVRISPmkII).Fixed a bug in SoftwareSerial (a hardware serial function was being called instead of the software serial equivalent). Thanks to brianbr for the report and fix.ARDUINO 0008 - 2007.06.09Updated examples (in distribution and on the website).Added an EEPROM library (see reference for details).Added a Stepper motor library (see reference).Patched to reduce binary sketch sizes by building the Arduino core as a library (.a) file - now only the needed parts of the core are linked into a sketch. Originally written by Nicolas Roland, revised by Don Cross.Fixed bug in Serial.available(). Report and fix by Don Cross.Now recompiling libraries when switching microcontrollers. Report by Julian Bleecker; fix by Nicholas Zambetti.Cleaned up core functions: moved pin definitions into program space to save RAM, and other minor improvements. Contributed by Jim Studt.Lots of reference additions and fixes from Paul Badger.Changed default microcontroller to ATmega168 from ATmega8.Removed the delay from analogRead().Activating TWI/I2C pullup resistors on the ATmega168 (in addition to the ATmega8).ARDUINO 0007 - 2006.12.25Smaller core (about 3.5 KB instead of 4.5 KB).Added a SoftwareSerial library (thanks to Antonio, Heather Dewey-Hagborg, and bigengineer for their help).Implemented a Serial.flush() routine; see reference for details.Digital pins 0 and 1 can be used for i/o until a call to Serial.begin().Replaced avr-lib's uart routines with custom code for handling serial communication and modified C++ serial commands to call the C serial commands; the code may behave slightly differently in border cases (e.g. non-standard speeds, or on overflow).Added attachInterrupt() and detachInterrupt() functions for handling of external interrupts on pins 2 and 3.Implemented shiftOut() routine; see reference for details.Defining binary constants: e.g. B1010 is 6.Mac versions no longer require running of the macosx_setup.command script.Windows version comes with the FTDI USB drivers already unzipped.New Linux binary distribution (still requires some programs to be pre-installed).ARDUINO 0006 - 2006.10.21Mac version no longer requires Java 1.5, meaning it should run on 10.3.9.Added support for analog inputs 6 and 7 and pwm on pins 5 and 6 on the on the ATmega168 used in the Arduino Mini (extra analog inputs not available in DIP ATmega168s).You now select the baud rate for the serial monitor from within the editor status bar when the serial monitor is running instead of from the Tools menu.Pressing enter within the serial monitor edit box no longer appends a newline to the message sent to the board.Included the Wire (TWI) library from Wiring.Updated the reference.ARDUINO 0005 - 2006.09.26Applied patch from Hans Steiner to improve Linux support by searching for avr tools in the user's path instead of expecting them at a fixed location.Added an upload.verbose preference for help in debugging.ATmega168 support!New Wiring-compatible randomSeed(), random(max) and random(min, max) functions (except operating on longs instead of floats).Fixed bug that sometimes caused uploading of old versions of a sketch.Serial monitor nows include an interface to send messages to the Arduino board. Pressing return appends a newline, pushing the send button doesn't.Now displaying "burning bootloader..." and "compiling..." status messages.ARDUINO 0004 - 2006.04.26Main sketch is now compiled as C++ (instead of C).Updated avr toolchain.printInteger(), printHex(), etc. now handle longs.millis() fixed (now overflows after days, not minutes)Fixed path to java in Windows run.bat.Added Matrix and Sprite libraries (written with Nicholas Zambetti).PWM now working on pin 11 (in addition to pins 9 and 10).Slowed PWM frequency (on all three PWM pins) to 1KHz.Now give an error if compiled sketch is too big.Fixed abs(), min(), max(), and constrain() macros.Added menu items to the IDE to burn bootloader.Now display binary sketch size on upload, and give error if too big.Added C++ serial library.Resynced with Processing/Wiring IDE code (improved auto-format, faster logging to serial monitor console, other bug fixes)New library system.Updated to latest version of the RXTX serial library; Mac users will need to rerun macosx_setup.command.ARDUINO 0003 - 2006.01.16API ChangesReversed the analog input pins to correspond to newer boards. This means a call, for example, to analogRead(0) must be changed to analogRead(5) in order to read the same physical pin.Added a printNewline() function (which sends '' = ASCII 10).New StuffReference is included (features native to C not yet documented).Serial monitor added (click the toolbar button to turn it on or off). Baud rate is controlled by the Serial Monitor Baud Rate Menu, defaults to 9600. Icon and implementation from Wiring.Serial port menu now automatically refreshes when opened.New blue color scheme and icons courtesy of Noah Shibley (colors are hardcoded into the source to ensure consistency with image files).Keyspan and FTDI USB drivers included with Mac and Windows distributions.Bug Fixesmillis() now updates every millisecond instead of every second.Bootloader included with Windows distribution (it was already in the Mac dist).Updated icon of the Windows executable.Now flushing the serial port before uploading (should fix some errors).Improved accuracy of the delayMicroseconds() function.OtherUpload rate no longer selectable from a menu within the IDE. Instead, edit the serial.download_rate item in the preferences.txt file.Created Xcode project for building Arduino on the Mac (doesn't yet regenerate the grammar files or package the distribution); active target should be "App".Removed unused or unimplemented items from menus.ARDUINO 0002 - 2005.10.05New build process no longer uses makefiles; now controlled by preferences.txt.core/ replaced with targets/; can now link against Wiring libraries.Replaced print() with printString, printInteger, printHex, printByte, etc.Added menu for selecting serial port speed.Updated icon.Bootloader shrunk to less than 1 KB; fuses updated accordingly.Added serialRead(), serialAvailable(), and delayMicroseconds().ARDUINO 0001 - 2005.08.25This is the first released of the unified IDE + language library it's a terrible hack... but it works. at the moment it's in alpha stage but it can be used to work.The processing preprocessor is included but not used. Created on: 5 January 2022 How to install the Arduino IDE software application for writing Arduino sketches. Install the Arduino IDE and load a test sketch to the target board. Software called the Arduino IDE (Integrated Development Environment) is used to load programs to Arduino boards. A program that runs on an Arduino board is called a sketch. Many example sketches come with the Arduino IDE. Modify these sketches to suit your application. Also write your own sketches from scratch. Part 2 of the Arduino Tutorial for Beginners Install the Arduino IDE software to a computer in this second part of the Arduino tutorial for beginners. It follows the first part of the Arduino tutorial for beginners. Use the Arduino IDE to load a sketch to an Arduino Uno, or an Arduino MEGA 2560. The following image shows the Arduino IDE application window. An example sketch is open in the Arduino IDE window in the image. How to Install the Arduino IDE Software Application Shown in this Image Download the Arduino IDE Software Go to the Arduino software downloads page. Find download links to the latest Arduino IDE near the top of the page. Details on installing the software follow in the next section. Windows, Linux and Mac computers all have a ZIP file version of the Arduino IDE. The Arduino IDE can optionally be installed using an executable download file on a Windows computer. Alternately, the Arduino IDE can be installed from the Microsoft app store. Leave the Arduino software downloads page open. Meanwhile read the next section for installation instructions. Install the Arduino IDE Software Instructions on how to install the Arduino IDE software follow for different operating systems. Information on getting started with the Arduino IDE is also provided by Arduino on the Getting Started with Arduino page. How to Install the Arduino IDE on a Windows Computer Two options for installing the Arduino IDE on a Windows computer follow. Either install from the executable, explained first below. Alternately, install using the ZIP file, explained in the second sub-section that follows. Executable Installer for Windows On a Windows computer, Arduino suggest installation using the executable installer (.exe). The reason for this is that it also installs the drivers. To use this installation method, click the top download option on the Arduino software downloads page, as shown in the following image. A red dot marks the executable download link in the following image. Windows Arduino IDE Download Options After clicking the link, a web page opens. On this page, either make a donation and download. Alternately, click the JUST DOWNLOAD button to download immediately. Locate the downloaded file using Windows file manager after the download finishes. At the time of writing this tutorial, the file name is arduino-1.8.19-windows.exe. The version number in this file name changes whenever the IDE software is updated. Therefore the file you downloaded may have a different name. Double-click the file to start the installation. Windows prompts with a dialog box that says: Do you want to allow this app to make changes to your device? Click the Yes button in the dialog box. In the next dialog box that opens, click the I Agree button to accept the License Agreement. The next dialog box is the Installation Options box. If you dont want a Desktop shortcut to be created, then uncheck this option. Otherwise, leave all of the options checked in this dialog box. Click the Next button. In the Installation Folder dialog box, leave the default destination folder as is. Finally click the Install button. It takes a few minutes for the installation to complete. A dialog box pops up at the end of the installation. It prompts with: Would you like to install this device software? Click the Install button. The same dialog box pops up for a second and third driver. Click the Install button in both instances. After the drivers have finished installing, click the Close button. The Arduino IDE software installation is now complete. See the Arduino IDE Windows installation guide for alternate installation instructions. ZIP File Windows Install An alternative to installing the Arduino IDE software using the executable installer on a Windows computer is to use the ZIP file. Simply click the Windows ZIP file link. The previous image shows this link. It is the second link, under the one marked with a red dot. Find this on the Arduino software downloads page again. After downloading the ZIP file, double-click it to open it. Drag the folder from the opened ZIP file to the Windows Desktop. This unzips the folder and places it on the Desktop. To run the Arduino IDE, first double-click the Arduino IDE folder on the Desktop to open it. Double-click the arduino.exe file in the folder to start the Arduino IDE. If file extensions for known file types are hidden, then this file will appear just as arduino, without the .exe at the end. Drivers may need to be manually installed later with this type of installation. For more information on making a portable version of the Arduino IDE software using the ZIP file, see the article on how to create and use a portable version of the Arduino IDE. This installs the Arduino IDE on a USB Flash drive. How to Install the Arduino IDE on a Linux Computer Download the correct Arduino IDE ZIP file from the Arduino software downloads page. Most modern Intel architecture computers will use the Linux 64-bit ZIP file. A red dot marks the Linux 64-bit ZIP file link in the following image. Linux 64-bit Arduino IDE ZIP Download Link After downloading the ZIP file, extract the folder inside. Copy it to the Linux Desktop or other convenient location. Double-click the arduino file in this folder to start the Arduino IDE. Add the user to the dialout group to be able to load Arduino programs, or sketches, from a Linux computer to the target Arduino board. Enter the following at the command prompt to add your Linux user name to the dialout group, where is your Linux login user name: sudo usermod -a -G dialout After adding your user name to the dialout group, either log out and back in again for the change to take effect. If there are any troubles loading a sketch to the target Arduino board, then try rebooting the computer. For an alternate Linux installation method that uses an installation script, see the Arduino Linux IDE installation guide. See how to create and use a portable version of the Arduino IDE for a portable Linux installation on a Flash drive. How to Install the Arduino IDE on a Apple Mac Computer Click the Mac OS link on the Arduino IDE software download page. This downloads a ZIP file version of the Arduino IDE for Apple Mac computers. Follow the Arduino Mac OS installation guide to complete the installation. Load a Sketch to the Arduino Board Test the installed Arduino IDE by loading a sketch to the target Arduino board. The target board is either an Arduino Uno or Arduino MEGA 2560 in this tutorial. The following steps describe how to load a sketch to the board. Connect the Arduino Board to the Host Computer Use a type A-B USB cable to connect the Arduino Uno or Arduino MEGA 2560 to the host computer. Host computer refers to the computer that the Arduino IDE is installed on. Genuine Arduino Uno and MEGA 2560 boards have an ON LED that lights up when the board is plugged into the USB port of a computer. Clone and compatible boards usually have a similar LED that indicates that the board is powered from USB. Start the Arduino IDE and Select the Board Start the Arduino IDE on the host computer. After that, select the Arduino board that you are using. This is either the Arduino Uno, or Arduino MEGA 2560 in this tutorial. Thereafter, select the port that the board is configured as. Board and port selection details follow. Arduino Uno Board Selection Select Tools Board Arduino Uno using the top menu of the Arduino IDE. The Arduino Uno may be found under Tools Board Arduino AVR Boards Arduino Uno, depending on the version of the IDE. This selects the Arduino Uno as the board to program. The port that the board is assigned must now be selected. See Arduino Port Selection that follows the next section. Arduino MEGA 2560 Board Selection Select Tools Board Arduino Mega or Mega 2560 using the top menu of the Arduino IDE. The Arduino MEGA 2560 may be found under Tools Board Arduino AVR Boards Arduino Mega or Mega 2560, depending on the version of the IDE. When Arduino MEGA is selected for the board, a new item appears on the Tools menu below Boards. This item is called Processor. Make sure that ATmega2560 (Mega 2560) is selected next to the Processor item. This selects the Arduino MEGA 2560 as the board to program. The port that the board is assigned must now be selected, as described next. Arduino Port Selection Select Tools Port using the top menu of the Arduino IDE. Where is the port number that the operating system assigned to the board. On windows this is a COM port. For example, COM4. The port could have a different number, such as COM3, COM5, and so on. On a Linux computer, the port may appear as /dev/ttyUSB0, /dev/ttyACM0, or similar. On a Mac the port could be /dev/cu.usbmodem14112 or similar. Some compatible Arduino boards dont show the Arduino board name next to the port number. If there is more than one port on the computer and it is not clear which one is the Arduino, then unplug the Arduino board. After that, see which port number disappears from the port list on the Tools menu. Plug the board back in, and the port that reappears in the port list is the Arduino board. Open an Example Sketch Select File Examples 01.Basics Blink on the top menu of the Arduino IDE. This opens the Blink example sketch in a new Arduino IDE window. When this sketch is loaded to the target Arduino board, it blinks or flashes the on-board LED marked L. Load the Example Sketch to the Arduino Board Click the Upload button in the Arduino IDE to load the Blink sketch to the target Arduino board. The Upload button is the right pointing arrow icon on the Arduino IDE toolbar shown in the following image. Arduino IDE Upload Button After clicking the Upload button, the sketch is built and uploaded to the target Arduino board. If everything is set up correctly and working, the L LED on the Arduino board starts to blink at a slow rate. Fault Finding If the sketch does not upload to the board, then make sure that all of the instructions in this part of the tutorial were followed. Check that the board is plugged into the USB port of the host computer, and that the ON LED lights up. Next, check that the correct board and port are selected in the Arduino IDE. Try rebooting the computer if there are still problems. On a Linux computer, make sure that you added your username to the dialout group. Log out and log back in after this. Other problems can be a faulty USB cable, or USB cable that is for charging only, with data not connected. During the making of this tutorial, the Arduino IDE did not upload to the Geekcreit clone Arduino MEGA 2560 shown in the previous part of this tutorial. An attempt to remedy this problem by burning the bootloader to the board using an external programmer failed when using the Arduino IDE. The solution that got the board working was to load the bootloader to the board using Microchip Studio. This procedure is described in the Ultimate Arduino Mega 2560 Hardware Manual. The Ultimate Arduino Mega 2560 Hardware Manual for the Arduino MEGA 2560, and the Ultimate Arduino Uno Hardware Manual for the Arduino Uno, both provide much more information on fault finding and testing these Arduino boards than can be covered in this tutorial. Next Part of this Arduino Tutorial for Beginners Now that you have discovered how to install the Arduino IDE software on a host computer, and have loaded an example sketch to the target Arduino IDE, you can move to the next part of this tutorial. The following parts of this Arduino tutorial for beginners rely on having a working Arduino IDE installed, and a working Arduino board. Previous:Arduino Tutorial for BeginnersNext:Arduino Example Sketches for Beginners

Install arduino ide mac. Install arduino ide on raspberry pi 5. Install arduino ide on chromebook. Install arduino ide 1.8.19. Install arduino ide ubuntu 22.04. Install arduino ide debian. Install arduino ide fedora. Install arduino ide windows. Install arduino ide for esp32. Install arduino ide linux mint. Install arduino ide linux. Install arduino ide on raspberry pi. Install arduino ide ubuntu. Install arduino ide ubuntu 20.04. Install arduino ide arch.