

Read more by scanning the QR code below



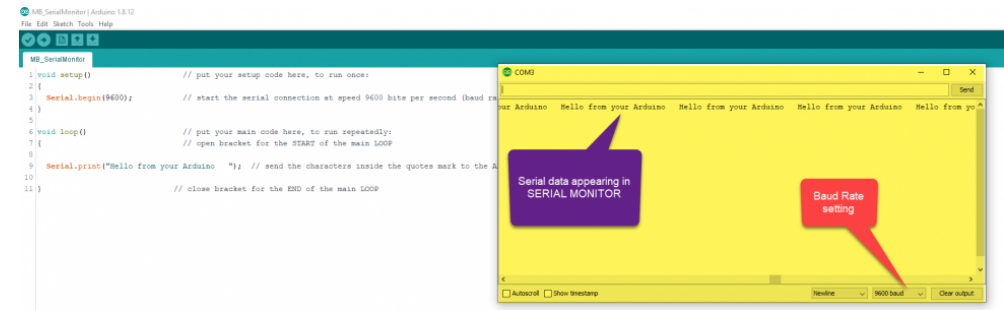
or by going to:

model-boats.com/77106



Arduino and model boats

by G6SWJ



23rd Jul 2020

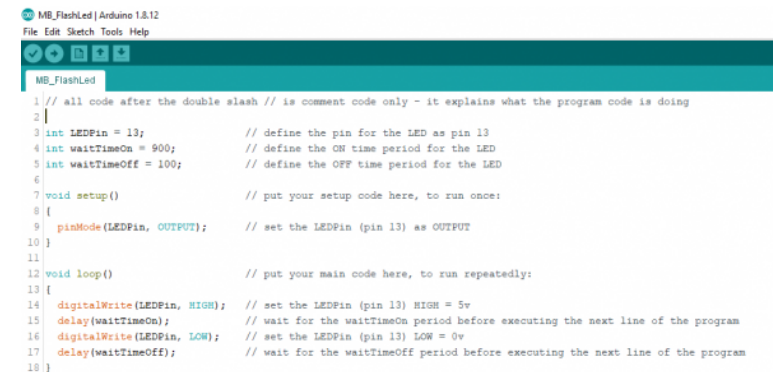
Arduino and model boats

PART ONE Hello – it's great to see that there are a bunch of you interested in Arduino and model boats I love playing with Arduino, I learn something new almost every day and find it mentally stimulating and rewarding working through the "How can I do this" of an Arduino project. It's a big topic and comes with a set of it's own jargon so it can be difficult to know where to start, how much is it going to cost, where can I purchase the boards etc. etc. There are some fundamental building blocks that you need in place to be able to start your Arduino journey and it can be all too easy to give up on the subject when you hit a brick wall. My approach will be blended learning – there are some fabulous free learning resources on the internet – these are top notch well thought out and well-presented short videos. We all have different levels of understanding about Arduino – I will pitch the posts at those just starting their adventure – read on past if this is not you and and you come across familiar territory and maybe check back in when things get a bit more juicy. A taster for some of the projects to come A bilge water alarm - <https://www.youtube.com/watch?v=HcpV0fIL-zA> A Morse code/ Aldis flasher - https://www.youtube.com/watch?v=8_SiZkciRsA No more waffle from me - lets scratch the surface of 2 of the building blocks – investing time watching these video a couple of times each will give you a good grounding in the subject. Arduino Hardware overview and Arduino Integrated Development Environment – the free text editor like software where you create a program (from this point onwards referred to as the IDE) The four videos below have some overlap but they give a great introduction - enjoy <https://youtu.be/nL34zDTPkcs> <https://youtu.be/09zfRaLEasY> <https://www.youtube.com/watch?v=fJWR7dBuc18> <https://www.youtube.com/watch?v=CSx6k-zXILE> This post is intended to set the scene and give you an overview of the subject Please do post any questions / feedback PART TWO will cover: Writing your first program Regards Jonathan

24th Jul 2020

Arduino & Model Boats - Part 2

If you have watched the "setting the scene" videos from Part 1 you should now be more familiar with some aspects of Arduino. I am going to suggest that you don't purchase any boards or accessories right now. Wait until we have covered a bit more of the topic – you will be able to judge then whether this is for you or not and therefore save wasting money. In Summary: > Arduino is open source and anybody can manufacture the boards – if not made by Arduino(Italy) then they should have a different name e.g. Seeeduino > The boards have inputs and outputs (Analog 0-5v, Digital 0 or 5v) > It is necessary to select the com (communication) "port" in the IDE that the Arduino board is connected to > It is necessary to select which Arduino board type (Uno,Nano etc) you are using in the IDE > Programs (also known as Sketches) are written in the free IDE computer software > Programs are uploaded from the IDE to the Arduino board using a USB cable The next video shows you how to write a very simple program to flash an LED - the full program code can be seen in the attached image https://www.youtube.com/watch?v=d8_xXNcGYgo Regards Jonathan



```
1 // all code after the double slash // is comment code only - it explains what the program code is doing
2
3 int LEDPin = 13;           // define the pin for the LED as pin 13
4 int waitTimeOn = 900;      // define the ON time period for the LED
5 int waitTimeOff = 100;     // define the OFF time period for the LED
6
7 void setup()               // put your setup code here, to run once:
8 {
9   pinMode(LEDPin, OUTPUT); // set the LEDPin (pin 13) as OUTPUT
10 }
11
12 void loop()                // put your main code here, to run repeatedly:
13 {
14   digitalWrite(LEDPin, HIGH); // set the LEDPin (pin 13) HIGH = 5v
15   delay(waitTimeOn);          // wait for the waitTimeOn period before executing the next line of the program
16   digitalWrite(LEDPin, LOW);  // set the LEDPin (pin 13) LOW = 0v
17   delay(waitTimeOff);         // wait for the waitTimeOff period before executing the next line of the program
18 }
```

24th Jul 2020

Arduino & Model Boats - Part 3

HINTS AND TIPS >>>DOCUMENT your code as you go we can add comments to our program code to help us understand what each line or section is doing. All text on the same line after a double forward slash is defined as a comment e.g. // This is a comment This has no effect on the program code or memory use when uploaded to the Arduino. In a simple program this may seem unnecessary – it is however a good habit to adopt. When you open a program 6 months down the road you may find you look at the code you wrote and struggle to understand your long forgotten logic – it can be very annoying when you can't understand your own code! >>SAVE your program regularly with a different name I use an approach where I give the program a name and suffix _xxx. xxx is simply an incremental serial number “MyLedFlasher_001.ino”, “MyLedFlasher_002.ino” etc Often my enthusiasm overruns and I change lots of lines of code all at the same time only to find the program does not do what I expected or results in an error. If you have overwritten your previous working copy with the same file name you will feel your blood pressure rising. When the going gets tough I rename after each and every change - trust me this approach pays dividends...>>SYNTAX – what is it? Definition = “the structure of statements in a computer language” There are a set of rules (syntax) you need to comply with in order for the IDE to accept your program code as valid. For example one syntax rule is most lines of code require the “line terminator character” – a semi colon to be added to each line of code to tell the IDE “end of line”. If you make Syntax errors then the code will not upload(compile) to the Arduino board and you will be left with an orange error bar at the bottom of screen and some error text hinting at what might be wrong with your code. Another example of a Syntax error is incorrect use of lower case or capital letters for program commands. digitalWrite(LEDPin, HIGH); is valid syntax and works digitalWrite(LEDpin, HIGH); is invalid syntax and does not work as the letter w is in lower case Brackets are always used in pairs - open and close () or {} - it is easy when code gets more complex to omit a bracket Get used to the orange error bar and syntax errors they happen all the time to all programmers – or at least it does for me. You will get familiar with solving the errors over time – some can take a few minutes to work out what you have done wrong. In the worst case it can be best to take a step backwards – open the previous saved working version, rename it so the working version is left untouched and then start again with your changes. -x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x- So that was some heavy stuff, on a lighter note looking forward to some of the upcoming projects I will touch on - very happy to add requests to this list Bilge water alarm Temperature Alarm Dimming LED's Morse code & Aldis lamp signalling your own message Simulating rotating emergency light Moving a servo - this could be to let down a ramp or simulating gun turret movement Slowing down servo movement Running a servo movement sequence Automatic ballast trim to correct/rebalance vessel leaning to one side or other Active stabilisers Reading the receiver channels Engine sound device Switching 16 circuits from just one transmitter stick Holding a compass heading Remember Rome was not built in a day - Be patient with yourself and enjoy this fascinating topic Regards Jonathan

11th Aug 2020

Arduino & Model Boats 8 - Bilge Water Alarm

Here is the full code for a Bilge Water Alarm. If you have followed this thread you should be able to follow what's going on - the video is using the same code as posted below. Part of learning Arduino is uploading someone else's working code that you may not understand to your own Arduino board and then changing values etc and seeing what affect that has

Regards Jonathan _ _ _ + + + + + // Water Alarm 25/07/2020 // Use 2 wire probes located in the area that may contain water, place them a few cm's apart // Connect one wire to a GND pin the other to Analog pin A0 // To make the sound we connect a small piezoelectric sounder to pin 6 and GND int soundFreq = 700; // Sound frequency (in Hz) int soundStep = 50; // Sound frequency increment - to make the alarm sound vary in pitch we will increment the tone frequency when it is triggered const int ledPin = 13 ; // LED pin const int soundPin = 6; // Ouput pin 6 connected to our piezoelectric sounder device const int sensorPin = A0; // Simple piece of wire connected to pin A0 used as a probe located in area that may contain water uint16_t sensorValue = 0; // variable to store the analog pin value when it is read using command analogRead(pin) - value will be in the range 0-1023 void setup() // Setup code - this runs once and sets up output etc. { pinMode(ledPin, OUTPUT); // define our ledPin as output pinMode(soundPin, OUTPUT); // define our soundPin as output pinMode(sensorPin, INPUT_PULLUP); // define our analog sensor pin as input - we are using pin A0 //digitalWrite(A0, INPUT_PULLUP); // Set the default state } void loop() { sensorValue = analogRead(sensorPin); // Read the analog value of A0 - will return a value between 0 and 1023 if (sensorValue > 1000) // test the analog sensor value now stored in variable "sensorValue" - if it's greater than 1000 then continue. If less than 1000 process the code defined by "else" { digitalWrite(ledPin, LOW); noTone(soundPin); delay(50); } else // if the sensoValue is less than 1000 then process the code below { digitalWrite(ledPin, HIGH); // Turn on the LED by setting the ledPin to High (5v) // The code chunk below is used to make the frequency of the sound on the soundPin vary // By increasing frequency to a preset value and then decreasing to a preset value - it then repeats each loop up & down in frequency to shape the alarm sound tone(soundPin, soundFreq); // Set the soundPin to the value stored in soundFreq soundFreq = soundFreq + soundStep; // increment the soundFreq by adding the value stored in the variable named soundStep - if soundStep = 50 the frequency will increase / if soundStep = - 50 the frequency will decrease if (soundFreq == 1200) // test the soundFreq - if it's reached 1200 then { soundStep = -50; // change value of soundStep to - 50 } if (soundFreq == 700) // test the soundFreq Value { soundStep = 50; // change value of soundStep to 50 } delay(25); // pause 25ms digitalWrite(ledPin, LOW); // turn the led pin off (0v) by setting the value to LOW delay(25); pause 25ms } } Below is the same code but without comments - might make easier reading as displayed on the forum. int soundFreq = 700; int soundStep = 50; const int ledPin = 13 ; const int soundPin = 6; const int sensorPin = A0; uint16_t sensorValue = 0; void setup() { pinMode(ledPin, OUTPUT); pinMode(soundPin, OUTPUT); pinMode(sensorPin, INPUT_PULLUP); //digitalWrite(A0, INPUT_PULLUP); } void loop() { sensorValue = analogRead(sensorPin); if (sensorValue > 1000) { digitalWrite(ledPin, LOW); noTone(soundPin); delay(50); } else { digitalWrite(ledPin, HIGH); tone(soundPin, soundFreq); soundFreq = soundFreq + soundStep; if (soundFreq == 1200) { soundStep = -50; } if (soundFreq == 700) { soundStep = 50; } delay(25); digitalWrite(ledPin, LOW); delay(25); pause 25ms } }

11th Aug 2020

Arduino & Model Boats 7

Apologies to those interested in this topic for the long gap between posts - life got in the way! Summary of learning points so far: > Arduino is open source – anybody can make the boards > Arduino boards have inputs and outputs - analog (0-5v) and digital 0 or 5v > We use the IDE to write our code and upload it via a USB cable to the Arduino board > We need to tell the IDE which com “port” and type of Arduino board we are using > Programs are also known as sketches > Programs have 2 parts – SETUP runs once as the program starts and LOOP that runs continually > DOCUMENT code as you create it using // This is a comment > SAVE your code regularly using a new name each time, the name hints at what the program does – e.g. MyLedFlasher_001, MyLedFlasher_002, etc > Syntax – a set of language rules you must comply with >> Terminate lines with ; >> Use upper and lower case appropriately where needed >> Brackets always used in pairs – open and close > We can send text from the Arduino to the IDE Serial Monitor popup window by using the command Serial.print() Arduino has INPUTS and OUTPUTS In this post we will use an input 'state' to send some text to the IDE Serial Monitor popup The code below monitors the 'state' of one input pin (pin8) and depending on the 'state' of pin8 (HIGH or LOW) sends some text to the IDE Serial Monitor popup window. To make pin 8 HIGH I connect a jumper cable between the Arduino 5v pin and pin 8 To make pin 8 LOW I connect a jumper cable between the GND (ground or zero volt pin) and pin 8. When using Arduino you will regularly come across the concept of pins (both INPUT and OUTPUT) being HIGH or LOW. HIGH simply means that the pin is at 5 volts and LOW that the pin is at 0 volts (zero or GND (ground)) – NB Some Arduino's are based around 3.3v logic not 5v – I'll cover this later So: HIGH = 5volts or logic value 1 (one) and LOW = 0 volts or logic value 0 (zero) The HIGH or LOW are often referred to as “logic states” To 'test' the logic state of the pin8 we will use the command digitalRead(8) – this will return one of two values – '1' = the pin is high @ +5v or '0' the pin is LOW @ 0v ++++++ We need a way to work with the result 1 (HIGH) or 0 (LOW) - the result of the digitalRead(8) To do this we use the 'if' command. The 'if' command checks for a condition (e.g. ==1) and executes some code only if the condition is 'true'. The code below will check if pin8 ==1 and if true print text to the IDE Serial Monitor void setup() // put your setup code here, to run once: { Serial.begin(9600); // start the serial communication pinMode(8, INPUT); // set the pin 8 as INPUT } void loop() // put your main code here, to run repeatedly: { if (digitalRead(8)==1) // test if pin8 is HIGH (value of 1) { // if the above condition is 'true' send the text // to the serial port which we can view in the IDE Serial Monitor Serial.println("Digital pin 8 is HIGH"); } } A Youtube video link is attached that shows the output text in the IDE Serial monitor - it's not very exciting! Well done if you have made it here !! we now have enough building blocks in place with some minor additions to do some more meaningful stuff.... Regards Jonathan

25th Jul 2020

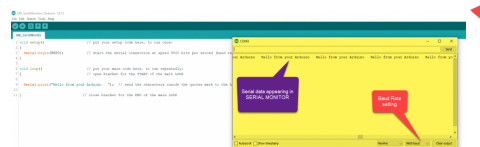
Arduino & Model Boats - Part 4

Summary of learning points so far: > Arduino is open source – anybody can make the boards > Arduino boards have inputs and outputs - analog (0-5v) and digital 0 or 5v > We use the IDE to write our code and upload it via a USB cable to the Arduino board > We need to tell the IDE which com “port” and type of Arduino board we are using > Programs are also known as sketches > Programs have 2 parts – SETUP runs once as the program starts and LOOP that runs continually > DOCUMENT code as you create it using // This is a comment > SAVE your code regularly using a new name each time, the name hints at what the program does – e.g. MyLedFlasher_001, MyLedFlasher_002, etc > Syntax – a set of language rules you must comply with >> Terminate lines with ; >> Use upper and lower case appropriately where needed >> Brackets always used in pairs – open and close Future topics we will cover: + Data types + Compiling code + Debugging code + Functions + Common programming structures/concepts e.g. "If this then do that" + Command sheets + Addin libraries + Interrupts + Serial monitor + Debouncing + i2c & SPI + PWM = pulse width modulation + Sensors + Arduino performance + How to power the Arduino + Quick glimpse at ATtiny, Teensy and esp32 And the command I hate the most “delay” - like driving a Ferrari with the hand brake on... Regards Jonathan

27th Jul 2020

Arduino & Model Boats 5

After this buidling block hopefully things will start to fall into place and become more interesting - hold the faith? The next building block we are going to cover is the IDE SERIAL MONITOR. The SERIAL MONITOR allows us to "see" what is going "under the hood" of our program – it's a bit like having a see-through oven door! to get used to how it works we will send a simple message from the Arduino to the IDE Serial Monitor on our desktop computer. The SERIAL MONITOR is mainly used during program development & debugging to help identify what is going on inside our program LOOP as will be seen in the next tutorials.. Using the SERIAL MONITOR we can: >>> Receive data from the Arduino serial port (serial = one character after the other) >>> Send DATA to the Arduino serial port. (which I am not going to cover here) The SERIAL MONITOR is accessed through the IDE command ribbon by selecting "TOOLS" and then "SERIAL MONITOR" - when selected a popup window opens on your computer which is the SERIAL MONITOR. (see pic 2) To send serial data from the Arduino to the SERIAL MONITOR we have to: In SETUP Start the serial connection using the command Serial.begin(speed) where speed for serial communication is one of the following Supported baud rates are 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 31250, 38400, 57600, and 115200 In the program LOOP We need to send data to the serial port - to do this we use this command Serial.print(); an example might be Serial.print("hello from your Arduino"); Everything between the quote marks will be sent to the Arduino serial port and be received by the IDE SERIAL MONITOR via the USB cable and displayed in the desktop IDE popup window. The serial monitor popup in image 3 is highlighted in yellow The example program is detailed in image 1 and also below void setup() // put your setup code here, to run once: { Serial.begin(9600); // start the serial connection at speed 9600 bits per second (baud rate) } void loop() // put your main code here, to run repeatedly: { // open bracket for the START of the main LOOP // send the characters inside the quotes mark to the Arduino serial port Serial.print("Hello from your Arduino "); // close bracket for the END of the main LOOP and with all the // comments removed void setup() { Serial.begin(9600); } void loop() { Serial.print("Hello from your Arduino "); } NOTES # We need to make sure that the baud rate speed we set in the program SETUP – Serial.begin(9600); matches the SERIAL MONITOR baud rate – we can change the Serial Monitor baud rate by selecting a value from a drop down list just to the right at the bottom of the monitor popup window – if they don't match you will either see lots of odd characters or nothing at all in your SERIAL MONITOR popup window # A good speed to use to start with for the Serial Communication baud rate is 9600. # The Arduino Uno and Nano only have one serial port. Other Arduino boards may have more serial ports and we would need to use a slightly different command as we would also refer to the serial port number e.g. SETUP serial2 begin(9600); and



28th Jul 2020

Arduino & Model Boats 6 - Digital Multi Voice Sound

Time for some carrots!!! Putting in the time and energy to learn a new skill is hard work, until you reach the tipping point where things begin to fall into place it can all seem a bit confusing, boring and pointless. Often initial motivation ebbs and the realisation of the new skill fades. So time for something inspirationalDIGITAL MULTI VOICE SOUND DEVICE So these videos are of a guys amazing project which might re-ignite some motivation. It's "Jedi" level so not for the next tutorial? There are many different sounds/ engine types that are available free and you can easily encode your own sound - yes it's in a truck but think how this concept could be transferred to a model boat - engine sound, horn, anchor, guns, sea gulls , general harbour noise Total cost excluding small amplifier and speaker about £10 Oh and did I mention that this was Jedi level - it's a very complex project as published, hundreds and hundreds of lines of code and does some crazy amazing stuff - most not relevant to model boats - extract the good bits and we have a simple sound project which performs as well as some commercial boards costing 5 to 20 times the price.... However complex the code this is the single command that makes the noise dacWrite(DAC1, Value); Enjoy these videos - I hope you find them inspirational... Jonathan
<https://youtu.be/xKvJeDONK5M?t=28> <https://youtu.be/qd8sOaW5Di4?t=29>
<https://youtu.be/kLBms7BGrdM?t=13>