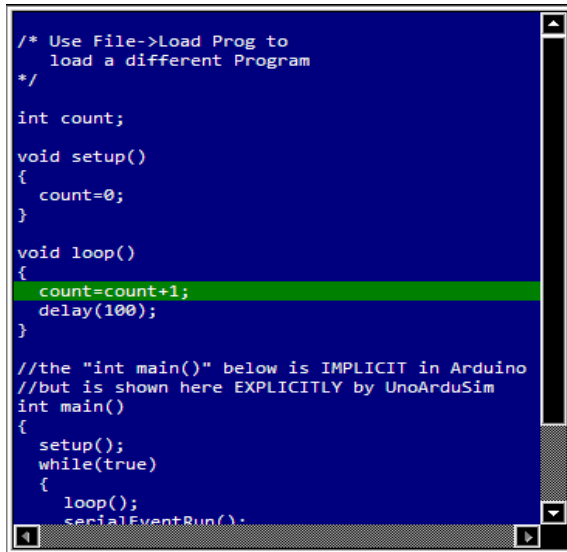


UnoArduSimV2.x Quick Help

The screenshot displays the UnoArduSimV2.x software interface, which is a virtual environment for simulating an Arduino Uno. The interface is divided into several main sections:

- Code Pane:** Located on the left, it shows the C++ code for a program. The code includes a loop that checks for a button press (pushPin) and then either steps a stepper motor forward or reverse, or checks an analog level (A2) and toggles a digital level (dirPin) based on a counter (tics).
- Variables Pane:** Located below the Code Pane, it displays the current values of variables defined in the code, such as backval, count, tics, digital_level, analog_level, numchars, and angle.
- Lab Bench Pane:** The central area of the interface, it features a detailed 3D model of the Arduino Uno board. Various components are labeled and interactive, including the ATMEGA328 microcontroller, push buttons, resistors (R=1K), a piezo buzzer, a stepper motor, a servo, a motor, and several LEDs (RYGB).
- Serial Monitor:** Located at the top right, it shows the data being sent and received via the serial port, including TX chars and Baud rate.
- Hardware Configuration:** The top right section contains controls for various hardware components like the stepper motor (steps, P1, P2), servo, motor, and LEDs.
- Signal Generators:** The bottom right section includes a PULSER and a FUNCGEN (function generator) for testing the board's response to different waveforms.
- Status Bar:** At the very bottom, it provides real-time feedback, such as "REACHED A Run TEMPORARY BREAKPOINT".
- Toolbar fly-over Hints:** A small box at the bottom left that provides additional information about the toolbar icons.

Code Pane:









```
/* Use File->Load Prog to
load a different Program
*/



int count;




void setup()
{
  count=0;
}



void loop()
{
  count=count+1;
  delay(100);
}

//the "int main()" below is IMPLICIT in Arduino
//but is shown here EXPLICITLY by UnoArduSim
int main()
{
  setup();
  while(true)
  {
    loop();
    serialEventRun();
  }
}
```

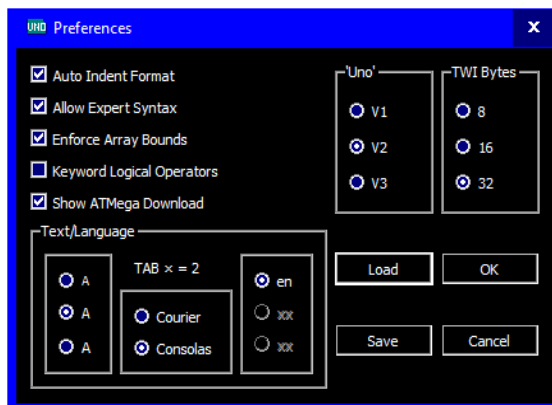
Step or Run using , , , or . To **Halt** at a specific program line, first click to highlight that line, and then click **RunTo** . To **Halt when a specific variable is written to**, first click on it to highlight it, and then click **RunTill** .

Jump between functions by clicking anywhere, then use **PgDn** and **PgUp** (or  and ).

Set search text with , and then **jump to that text** using  and .

Move between '#include' files using  .

Preferences:



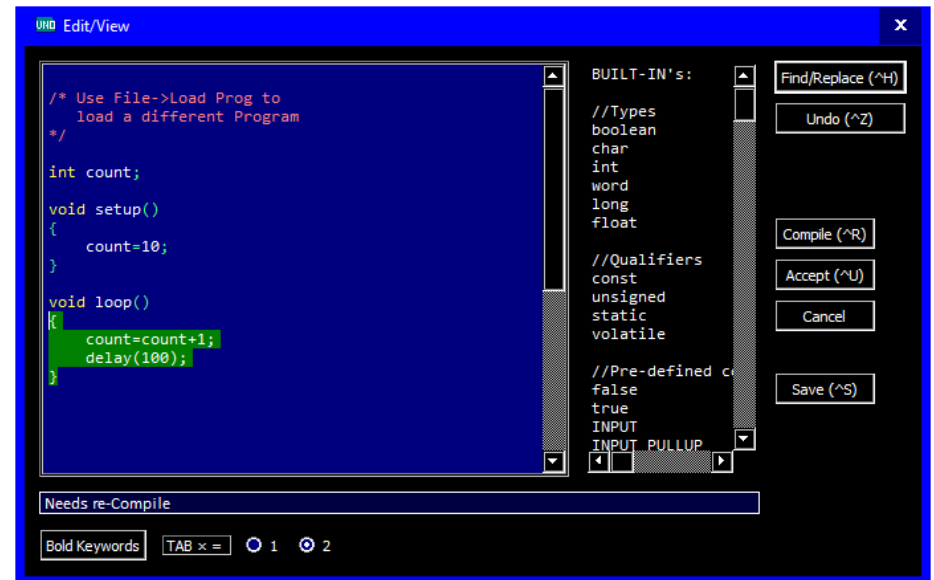
Configure→Preferences to set, save, and load user choices.

Alternate language(s) set by the user locale, and by a two-letter code on the very first line of the **myArduPrefs.txt** Preferences file

Edit/View:

To open at a specific line, **double-click** on that line in the **Code Pane** or use **File→Edit/View** (and it opens at the last highlighted line)

Tab-indentation will be automatically done if that preference is chosen from **Configure→Preferences** – you can also single or double-size the Tab width.



Add or delete tabs to a group of lines using **right-arrow** or **TAB**, and **left-arrow** (after first selecting a group of 2 or more consecutive lines).

To **add an item** (after the caret) **from the right-hand list of Built-ins**, double click on it.

Find (use ctrl-F), **Find/Replace** (use ctrl-H), **Undo** (ctrl-Z), **Redo** (ctrl-Y)

Compile and leave open (ctrl-R), or **Accept** (ctrl-U) or **Save** (ctrl-S) to close.

Find a brace's **matching brace-pair** partner by double-clicking on it – both braces, plus all text between, become highlighted (as in the image above).

Use **ctrl-PgDn** and **ctrl-PgUp** to jump to next (or previous) empty-line break.

Variables Pane:

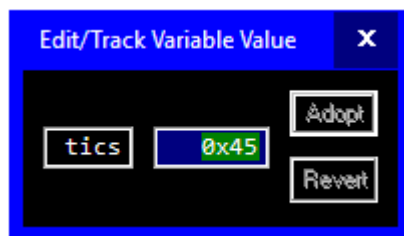
```
angle= 45
i= 8
k= 6
notefreq= 1046
dur= 0.12500
beats= 160
wholenote= 1500
quarternote= 375
msecs= 187
RingTones[0](-)
  RingTones[0](-)
    RingTones[0].frequency= 1046
    RingTones[0].duration= 0.12500
```

Click on (+) to expand, or on (-) to collapse arrays and objects.

PgDn and **PgUp** (or  and ) allow you to quickly jump between variables.

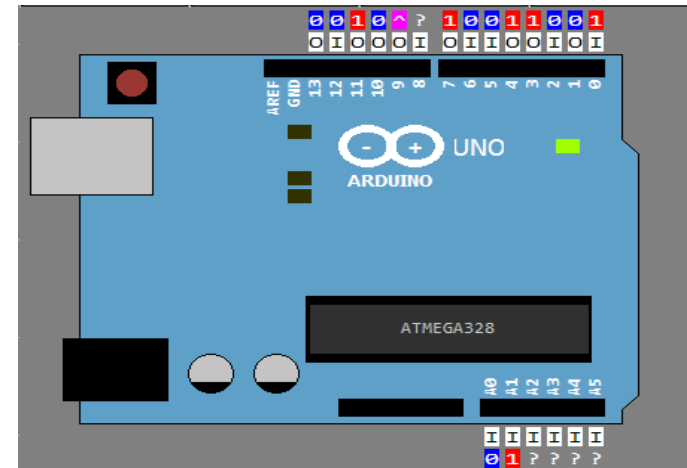
Use the **VarRefresh** menu to control update frequency when executing.

Double-click on any variable to track its value during execution, or to change it to a new value in the middle of (halted) program execution:

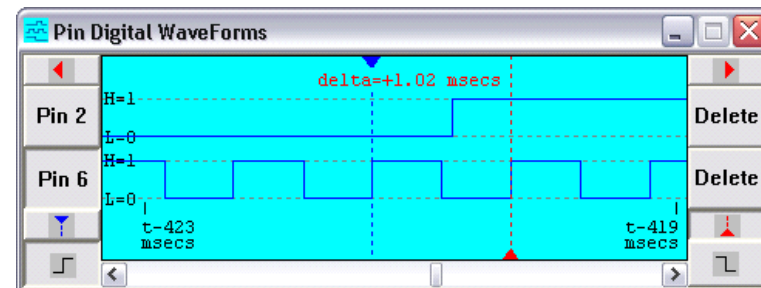


Or **single-click** to highlight any variable (or object-member, or array-element), then use **RunTill** to advance execution up to the next **write-access** to that variable or location.

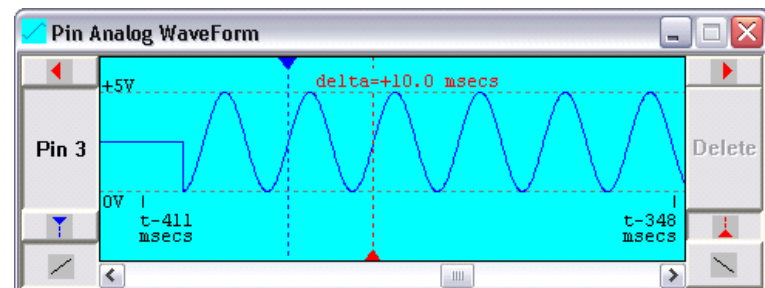
Lab Bench Pane and the 'Uno':



Left-click on any pin to create (or add to) Pin Digital WaveForms:



Right-click on any pin to create a Pin Analog WaveForm window:



To **ZOOM IN** and **ZOOM OUT** use the mouse wheel, or shortcuts **CTRL-up-arrow** and **CTRL-down-arrow**.

Lab Bench Pane 'I/O' Devices

Set numbers and types of each using **Configure→ 'I/O' Devices** . Set pins using a 2-digit value from 00 to 19 (or A0-A5).

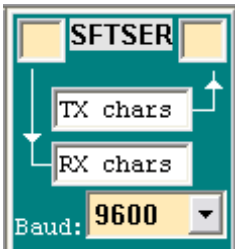
'Serial' Monitor ('SERIAL')



Type one or more characters in the upper ('TX chars') edit box and **hit Return**.

Double-click to open **a larger window for TX and RX characters**.

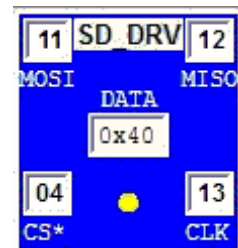
Software 'Serial' ('SFTSER')



Type one or more characters in the upper ('TX chars') edit box and **hit Return**.

Double-click to open **a larger window for TX and RX characters**.

SD Disk Drive ('SD DRV')

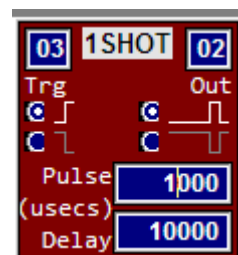


A small 8-Mbyte SD disk driven from SPI signals, and mirrored in an '**SD**' **subdirectory** in the **loaded program's** directory (which will be created if absent)

Double-click to open **a larger window to see Directories, Files, and content**

CS* low to activate.

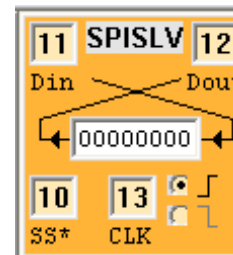
One-Shot ('1SHOT')



A digital one-shot. Produces a pulse of chosen polarity on '**Out**' after a specified delay from either a rising or a falling triggering edge seen on its **Trg** input. Once triggered, it will ignore subsequent trigger edges until the pulse on '**Out**' has been fully completed.

'Pulse' and '**Delay**' values (if suffixed with an 'S'). will be scaled from the toolbar 'I/O ____ S' slider

Shift Register Slave ('SRSLV')

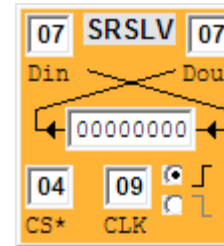


A simple shift-register device.

Edge transitions on CLK will trigger shifting.

SS* low, drives MSB onto Dout.

SPI Slave ('SPISLV')

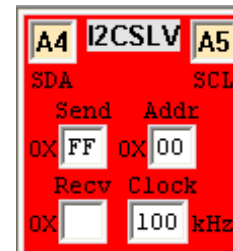


A mode-Configurable SPI slave device ('MODE0','MODE1','MODE2',or 'MODE3')

Double-click to open **a larger window** to set/view hex '**DATA**' and '**Recv**' bytes.

SS* low, drives MSB onto MISO.

Two-Wire I²C Slave ('I2CSLV')



A *slave-mode-only* I2C device.

Double-click to open **a larger window to set/view hex 'Send' and 'Recv' bytes**

Stepper Motor ('STEPR')

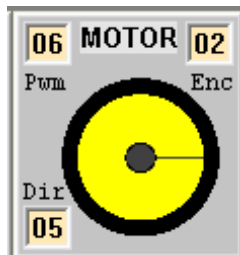


Accepts control signals **on either 2 or 4 pins. 'Steps' must be a multiple of 4.**

Use `'#include <Stepper.h>'` .

To emulate gear reduction by N in your program , use a modulo-N counter to determine when to actually call '`Stepper.step ()`'

DC Motor ('MOTOR')



Accepts PWM signals on **Pwm** pin, level signal on **Dir**, and outputs 8 highs and 8 lows per wheel revolution on **Enc**.

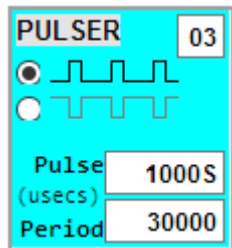
Full speed is approximately 2 revs per second.

Servo Motor ('SERVO')



Accepts pulsed control signals on specified pin. Can be modified to become continuous-rotation by checking the lower left check-box

Digital Pulser ('PULSER')

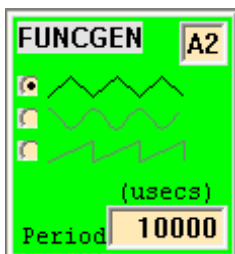


Generates digital waveform signals on specified pin.

Minimum period is 50 microseconds, minimum pulse width 10 microseconds. Both values (if suffixed with an 'S'). will be scaled from the toolbar 'I/O____S' slider

Choose positive-going pulses (0 to 5V) or negative-going pulses (5V to 0V).

Analog Function Generator ('FUNCGEN')



Generates analog waveform signals on specified pin.

Minimum 'Period' is 100 microseconds, scaled from the toolbar 'I/O____S' slider (if suffixed with an 'S').

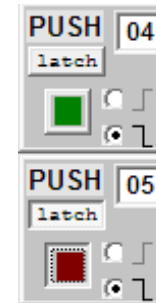
Sinusoidal, triangular, or sawtooth waveforms.

Piezoelectric Speaker ('PIEZO')



"Listen" to signals on any chosen 'Uno' pin.

Push Button ('PUSH')

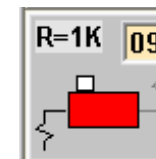


A normally-open **momentary** push-button to +5V or ground

A normally-open **latching** push-button to +5V or ground (depress "latch" button too get this mode).

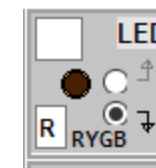
You can close the push-button by clicking it. or by pressing any keyboard key – contact bouncing will only be produced if you use the **space-bar** key.

Slide Switch Resistor ('R=1K')



A 1 k-Ohm pull-up to +5V OR a 1 k-Ohm pull-down to ground.

Coloured LED ('LED')



R,Y,G, or B LED connected between any chosen 'Uno' pin and either ground or +5V.





Analog Slider

A slider-controlled potentiometer. 0-5V to drive any chosen 'Uno' pin.



Menus






File:

<u>Load INO or PDE Prog</u> 	Allows the user to choose a program file having the selected extension. The program is immediately parsed
<u>Edit/View</u>	Opens the loaded program for viewing/editing.
<u>Save</u> 	Save the edited program contents back to the original program file.
<u>Save As</u>	Save the edited program contents under a different file name.
<u>Next (#include)</u> 	Advances the Code Pane to display the next '#include' file
<u>Previous</u> 	Returns the Code Pane display to the previous file
<u>Exit</u>	Exits UnoArduSim.









Configure:

<u>'I/O' Devices</u>	Choose desired number of each type of device (8 large, and 16 small, 'I/O' devices are allowed)
<u>Preferences</u>	Choose automatic indentation, font typeface, optional larger type size, expert syntax, keyword logical operators, enforcing array bounds, showing download, 'Uno' board version, and TWI buffer length

Find :

<u>Find Next Function/Var</u> 	Jump to the next Function in the Code Pane (if it has the active focus), or to the next variable in the Variables Pane (if instead it has the active focus).
<u>Find Previous Function/Var</u> 	Jump to the previous Function in the Code Pane (if it has the active focus), or to the previous variable in the Variables Pane (if instead it has the active focus).
<u>Set Search Text (ctrl-F)</u> 	Activate toolbar Find edit box to define your next-to-be-searched-for text..
<u>Find Next Text</u> 	Jump to the next Text occurrence in the Code Pane (if it has the active focus), or to the next Text occurrence in the Variables Pane (if instead it has the active focus).
<u>Find Previous Text</u> 	Jump to the previous Text occurrence in the Code Pane (if it has the active focus), or to the previous Text occurrence in the Variables Pane (if instead it has the active focus).

Execute:

<u>Step Into (F4)</u>		Steps execution forward by one instruction, or <i>into a called function</i> .
<u>Step Over (F5)</u>		Steps execution forward by one instruction, or <i>by one complete function call</i> .
<u>Step Out Of (F6)</u>		Advances execution by <i>just enough to leave the current function</i> .
<u>Run To (F7)</u>		Runs the program, <i>halting at the desired program line</i> -- you must first click to highlight a desired program line before using Run To.
<u>Run Till (F8)</u>		Runs the program, <i>halting when the highlighted Variables Pane variable location is next written to</i> (click to highlight a desired item before using RunTill).
<u>Run (F9)</u>		Runs the program.
<u>Halt (F10)</u>		Halts program execution (<i>and freezes time</i>).
<u>Reset</u>		Resets the program (all value-variables are reset to value 0, and all pointer variables are reset to 0x0000).
<u>Animate</u>		Automatically steps consecutive program lines <i>with added artificial delay</i> and highlighting of the current code line.
<u>Slow Motion</u>		Slows time by a factor of 10.

Options:

<u>Step Over Structors/Operators</u>	Fly right through constructors, destructors, and operator overload function during any stepping (i.e. it will not stop inside these functions).
<u>Register-Allocation Modelling</u>	Assign function locals to free ATmega registers instead of to the stack..
<u>Added loop() Delay</u>	Add 1 millisecond. (by default) to each call to <code>loop()</code> (in case user has not added any delays anywhere)
<u>Error on Uninitialized</u>	Flag as a Parse error anywhere your program attempts to use a variable without having first initialized its value.
<u>Show Program Download</u>	Show program download to the 'Uno' board (with attendant delay).

Configure menu commands:

<u>'I/O' Devices</u>	Choose desired number of each type of device (8 large, and 16 small, 'I/O' devices are allowed)
<u>Preferences</u>	Choose automatic indentation, font typeface, optional larger type size, expert syntax, keyword logical operators, enforcing array bounds, showing download, tab size multiplier, 'Uno' board version, TWI buffer length

VarRefresh:

<u>Allow Auto (-) Collapse</u>	Allow UnoArduSim to collapse displayed expanded arrays/structs/objects when falling behind real-time (<u>Allow Reduction</u> must also be set).
<u>Allow Reduction</u>	Allow reduced frequency of display updates in the Variables Pane to avoid flicker or reduce CPU load when falling behind real-time– then values shown are only updated periodically, <i>but also whenever the program is halted.</i>
<u>Minimal</u>	Only refresh the variables Pane display 4 times per second.
<u>HighLight Updates</u>	Highlight the last-changed variable value (can cause slowdown, and is cleared at each Reset).

Help menu commands:

<u>Quick Help File</u>	Opens the UnoArduSim_QuickHelp PDF file.
<u>Full Help File</u>	Opens the UnoArduSim_FullHelp PDF file.
<u>Bug Fixes</u>	View significant bug fixes since the previous release..
<u>Changes/Improvements</u>	View significant changes and improvements since the previous release.
<u>About</u>	Displays version, copyright

Windows:

<u>'Serial' Monitor</u>	Add a serial IO device (if none) and pull up a larger 'Serial' monitor TX/RX text window.
<u>Restore All</u>	Restore all minimized child windows.
<u>Pin Digital Waveforms</u>	Restore a minimized Pin Digital Waveforms window.
<u>Pin Analog Waveform</u>	Restore a minimized Pin Analog Waveform window.