# Monitor and modify variables with WawiLib

# CONTENTS

1	Intr	oduction2
	1.1	Objective of this document2
	1.2	Software and hardware requirements2
	1.3	Required user experience2
2	THE	"WawiWatchUSB" Demo sketch example3
3	The	WawiLib variable grid5
	3.1	Declaration type of a variable can differ from its displayed format5
	3.2	The column "Interface/Arduino ID"7
	3.3	The column "Arduino variable name"7
	3.4	The columns "Actual value" and "Format"7
	3.5	The column "Write value"9
	3.5.	1 Write bit format (BIT)11
	3.5.	2 Write hexadecimal format (HEX)11
	3.5.	3 Write integer format (INT/UINT)12
	3.5.	4 Write floating point format (FLOAT / DOUBLE)12
	3.5.	5 Write character format (CHAR)13
	3.5.	6 Write string format (STRING)15
	3.5.	7 Writing arrays of values16
4	Furt	her reading17



# 1 Introduction

## 1.1 Objective of this document

The objective of this document is to describe how to use WawiLib to monitor and modify variables of different types, formats and sizes.

WawiWatchUsb.ino, a demo sketch supplied with the WawiSerialUsb library, will be used to explain the concepts. The example sketch declares a number of variables and arrays of variables. WawiLib will be used to observe and modify these variables on your PC.

You may ask yourself: Why this demo? The user's interface of WawiLib is quite simple. You will discover possibilities in this document that are not documented in the getting started tutorials for USB, Ethernet and WiFi.

## 1.2 Software and hardware requirements

The Arduino IDE (in this example 1.8.15) and WawiLib V2.0.x both need to be installed on your PC. The demo runs with licensed and unlicensed versions of WawiLib. During the grace period of 2 months, you can test and use all functions without registration. After this period registration is required in order to access all functions. At this time registration is free. In the future a small contribution might be required to register in order to support the website.

WawiLib supports multiple interface types: serial, software Serial, USB, USB-native, TCP/IP, UDP/IP via Ethernet or WiFi. In this demo, the USB programming port of the Arduino is used as the communication interface between WawiLib and the Arduino shield.

The hardware you need is an Arduino board, a USB programming cable and a Windows PC (32 or 64 bit). In this demo, we will use the Arduino MEGA board but other boards can be used in a similar or even identical way. For compatibility of boards, go to <u>www.sylvestersolutions.com</u>.

## 1.3 Required user experience

The concepts of this document build further on the tutorial "*Getting started WawiLib programming port*". Some knowledge of the C programming language, especially how variables are represented in memory, is an advantage.

# 2 THE "WawiWatchUSB" Demo sketch example

Many of the Arduino libraries come with examples. WawiLib is not an exception. In this demo, we will use the sketch called WawiWatchUsb.ino

- ✓ Open the demo using the menu File\Examples\WawiSerialUsb\WawiWatchUsb in the Arduino IDE.
- ✓ Compile and download the sketch to your board.

```
#include <WawiSerialUsb.h>
WawiSerialUsb WawiSrv;
// test variables for demo:
bool demoBool = true;
char demoChar = 'C';
unsigned char demoUChar = 46;
int demoInt = -10;
unsigned int demoUInt = 12;
long demoLong = 0x1212;
unsigned long demoULong = 0x1313;
float demoFloat = 3.14152;
double demoDouble = 3.14152;
bool demoBoolAr[10] = { 0,0,1,1,0,1,0,1,1,1 };
char demoCharAr[25] = "Hello world.";
unsigned char demoUCharAr[10];
int demoIntAr[10];
unsigned int demoUIntAr[10];
long demoLongAr[10];
unsigned long demoULongAr[10];
float demoFloatAr[10];
double demoDoubleAr[10];
// make variables of interest known to WawiLib:
// this function is used in WawiSrv.begin(....)
void wawiVarDef()
{
    WawiSrv.wawiVar(demoBool);
    WawiSrv.wawiVar(demoChar);
    WawiSrv.wawiVar(demoUChar);
    WawiSrv.wawiVar(demoInt);
    WawiSrv.wawiVar(demoUInt);
    WawiSrv.wawiVar(demoLong);
    WawiSrv.wawiVar(demoULong);
    WawiSrv.wawiVar(demoFloat);
    WawiSrv.wawiVar(demoDouble);
    WawiSrv.wawiVarArray(demoBoolAr);
    WawiSrv.wawiVarArray(demoCharAr);
    WawiSrv.wawiVarArray(demoUCharAr);
    WawiSrv.wawiVarArray(demoIntAr);
    WawiSrv.wawiVarArray(demoUIntAr);
    WawiSrv.wawiVarArray(demoLongAr);
    WawiSrv.wawiVarArray(demoULongAr);
    WawiSrv.wawiVarArray(demoFloatAr);
    WawiSrv.wawiVarArray(demoDoubleAr);
}
```

```
void setup()
{
    Serial.begin(115200);
    WawiSrv.begin(wawiVarDef, Serial, "My Arduino");
}
void loop()
{
    WawiSrv.loop();
}
```

The demo sketch contains a series of variables declared as (static) variables in the sketch. For each variable type an array example is included in order to demonstrate the use of arrays with WawiLib.

You will see that all these variables are declared as static variables (e.g. outside of {}). Why? WawiLib determines the address and size of a variable once by calling the function wawiVarDef() when a new variable is added in the variable table of WawiLib-PC. From then on, it remembers the address and size of a variable. If a variable would be declared locally in a function, it would be created and destroyed on the stack every loop. So, visualizing such a variable does not make a lot of sense because a real time application cycles through its main loop very fast and continuously in order to keep all services running (other types of approaches for automation application are possible but rarely used).

All data traffic is done in the WawiSrv.loop() and WawiSrv.delay() functions. Therefore, make sure that these functions are called without large interruptions. If you have a local loop, you might consider calling WawiSrv.loop() additionally in that loop. If you want to add a delay to your sketch do not call delay() but call WawiSrv.delay() instead. WawiSrv.delay() splits the entire waiting interval in small fragments and calls WawiSrv.loop() and delay() repeatedly. In this way, communication with WawiLib on the PC remains active and you get a swift response form WawiLib on the PC.

How does the communication between WawiLib and the Arduino board work?

The PC is the master of the communication. The Arduino is the slave. The PC sends requests to investigate variable memory addresses, to read bytes at memory addresses or to write bytes to other memory addresses. The Arduino replies with the values, addresses or acknowledgments depending on the request issued.

Starting from WawiLib 1.6.0, the Arduino can also send messages to the PC on its own initiative, this mechanism is used to support the WawSerialXXX.print() function.

During serial communication, the maximum message size is determined by the Arduino core Libraries and the buffer sizes in the WawiLib library objects. If you run into limits related to your maximum serial message size, increment the maximum length of the Arduino Serial library and WawiLib will use these values as default settings. The default length of 64 bytes should be sufficient in most cases.

In order to increase performance, you can also increase the buffers inside the WawiSerialUsb object in the Arduino, this can be done by overriding the default values during the construction of the WawiSerialUsb object:

```
WawiSerialUsb WawiSrv(/*RX_BUF*/128,/*TX_BUF*/128,/*PR_BUF*/ 128);
or
WawiSerialUsbLight WawiSrv(/*RX_BUF*/128,/*TX_BUF*/128);
```

WawiSerialUsbLight is the light weight version of WawiLibUsb, it is smaller as it does not support .print() statements to send info from your sketch to the WawiLib-PC output window. WawiSerialUsbLight does not support breakpoings.

The message buffers within WawiSerialUsb (and its counterparts for WiFi and Ethernet) can be larger than the internal Arduino buffers for a protocol. For example an Arduino Serial buffer on a Mega 2560 is typically 64 bytes. Data telegrams are split into packages by the software if the internal Arduino buffers are too small compared to the WawiLib message buffers. During initialization, WawiLib-PC reads the sizes of the buffers on the Arduino side to make sure that Arduino and PC are on the same page. Both Arduino protocol and WawiLib object buffers are investigated during startup.

# 3 The WawiLib variable grid

# 3.1 Declaration type of a variable can differ from its displayed format.

In the Arduino processor, all variables are stored as series of bytes in memory. Whether you declare a variable as *char* or as *float*, both of them are stored as a number of bytes in the Arduino memory. The difference between *float* and *char* is very relevant once you compile the Sketch because the compiler needs to know the type of variable in order to translate your source code to the right CPU instructions.

WawiLib reads the byte(s) directly in the memory of the Arduino and puts them in the grid table. The type of formatting is determined by what you select in the WawiLib "Format" column. So, a char variable can be displayed as an *integer* and an *integer* variable can be displayed as a series of characters.

- ✓ Start WawiLib on your PC and fill in the table as below.
- ✓ Press "Setup()"
- ✓ You can fill in the variable names manually or use drag & drop from the tree.

File Ec	lit Settings	Help	-							3						
2 New	Open	Save	en Print	Copy	K Cut	Paste	Offli	ne Setup()	Loop()	Write	<b>≥ all</b> Continu∈	brkpt	⊠ brkpt			
- Avail	able variable	es	^	copy	Inter	face/Ard.		ariable nam	Actual	value	Write value	Format	bridge	Variabi	e address	and
🖻 se	r1\My Ardui	ino								10						
¢	Single varial	ble		1	ser1,	My Arduir	10	demoChar	0x	43		HEX	@demoChar=0	)x0238 [1	bytej x 1	VA
	demoBoo	bl		2	_	ser1		demoChar	6	7		INT	@demoChar=0	)x0238 [1	byte] x 1	VA
	🗄 demoCha	ar		3	_	ser1		demoChar	6	7		UINT	@demoChar=0	)x0238 [1	byte] x 1	VA
	demoUC	har		4		ser1		demoChar	(	1		CHAR	@demoChar=0	)x0238 [1	byte] x 1	VA
	demoInt			5		ser1		demoChar	0b010	0'0011		BIT	@demoChar=0	)x0238 [1	byte] x 1	VA
	demoUln	t		6		ser1		demoChar	?	?		FLOAT	@demoChar=0	)x0238 [1	byte] x 1	VA
	demoLon	g		7		ser1		demoChar	?	?		DOUBLE	@demoChar=0	)x0238 [1	byte] x 1	VA
	demoULo	ong		8		ser1		demoChar	STR FN	AT ERR		STRING	@demoChar=0	0x0238 [1	byte] x 1	VA
	demoFloa	at		9												
	demoDo	uble		10												
	Array			11												
	demoBoo	Ar[09]		12												_
	demoCha	rAr[024]		13												
	demoUC	harAr[09]		14												
	demoInt/	Ar[09]		47												
	damallin	10 01- 4+	~	<												>
Index	Time			Node		M	essage									1
000	29/07/202	1 10:19:16.	447	ser1/CON	22/My A	rduino Cl	osing c	ommunicatio	ns port							- 1
001	29/07/202	1 10:19:17.	005	ser1/COM	22/My A	rduino O	pening	serial port [C	DM22]	OK						
<																>

Fig 3.1. WawiLib with display of *demoChar* in various formats.

- ✓ Reduce the width of the "Recorder" column to 0 by dragging the right-hand column to the left.
- ✓ => Note: there is a "Reset Vieuw" option available in case you are lost in the UI.

	0.40		
3	Available interfaces	>	
3	Display formats	>	
3	Offline		
3	Setup()		
~	Loop()		
3	Write row	Ctrl+W	
-	Write selected fields		
•	Write all	Alt+W	
2	Cut	Ctrl+X	
2	Сору	Ctrl+C	
	Paste	Ctrl+V	
1	Clear row(s)		
-	Insert row		
	Insert rows		
-	Delete row(s)		
	Clear entire table		
	Variable properties		
£	Reset View	2	

Fig 3.2. Reset view option of WawiLib user's interface

In this example, the same USB interface is used for programming the Arduino and to communicate with WawiLib. Therefore, you cannot download your sketch to your board if WawiLib is online. In the same way, if you open a serial output window in the Arduino IDE, the Arduino IDE will claim the USB interface of your board and WawiLib will no longer be able to go online. So: during download, WawiLib needs to be offline in order to release the USB programming interface and vice versa.

If you want Arduino-IDE serial output and WawiLib communication at the same time, you can use another (serial or other) interface on your board for WawiLib. There are a number of USB to serial converters on the market. These interface converters can convert Arduino TTL logic to serial USB signals. The USB interface converter plugs into one of your PC's USB ports.

The Mega 2560, for example, has 3 additional serial interfaces. SoftSerial and USB-native (Arduino DUE) are also supported by WawiLib. There is a tutorial explaining the details to be taken into account using USB to serial converters on <u>www.sylvestersolutions.com</u>.

In fig. 3.3, you see the WawiLib screen after filling in the grid wit a variety of variables. You see that the default interface was added and the actual values are displayed in the selected format. On the right-hand side, you see the address of the variable, its size and the array count. In the last column, you see the status of the last variable read and recording operation.

	Interface/Ard. ID	Variable name	Actual value	Write value	Format	Variable address and status
1	ser1/My Arduino	demoChar	0x43		HEX	@demoChar=0x0238 [1 byte] x 1 VAR_READING_OK -
2	ser1	demoChar	67		INT	@demoChar=0x0238 [1 byte] x 1 VAR_READING_OK -
3	ser1	demoChar	67		UINT	@demoChar=0x0238 [1 byte] x 1 VAR_READING_OK -
4	ser1	demoChar	С		CHAR	@demoChar=0x0238 [1 byte] x 1 VAR_READING_OK -
5	ser1	demoChar	0b0100'0011		BIT	@demoChar=0x0238 [1 byte] x 1 VAR_READING_OK -
6	ser1	demoChar	??		FLOAT	@demoChar=0x0238 [1 byte] x 1 VAR_READING_OK -
7	ser1	demoChar	??		DOUBLE	@demoChar=0x0238 [1 byte] x 1 VAR_READING_OK -
8	ser1	demoChar	STR FMT ERR		STRING	@demoChar=0x0238 [1 byte] x 1 VAR_READING_OK -
9						
10	ser1/My Arduino	demoBoolAr[09]	0;0;1;1;0;1;0;1;1;1		INT	@demoBoolAr=0x0219 [1 byte] x 10 VAR_READING_OK -
11	ser1/My Arduino	demoLongAr[1]	0x00000000		HEX	@demoLongAr=0x03B6 [4 byte] x 10 VAR_READING_OK -
12	ser1/My Arduino	demoLongAr[2]	0			@demoLongAr=0x03B6 [4 byte] x 10 VAR_READING_OK -
13	ser1/My Arduino	demoCharAr[024]	Hello world.		STRING	@demoCharAr=0x0200 [1 byte] x 25 VAR_READING_OK -
14	ser1/My Arduino	demoFloatAr[2]	0		FLOAT	@demoFloatAr=0x0366 [4 byte] x 10 VAR_READING_OK -
15						

Fig 3.3. WawiLib grid with a variety of variables and arrays of variables.

# 3.2 The column "Interface/Arduino ID"

In the column "Interface/Arduino ID", you can see the name of the interface and the name of the board. If you have only 1 board connected to WawiLib, this column will be automatically filled in when you go online. If not, you can fill in the name of the interface manually (ser1 in this case) or click right and select the right interface from the menu.

If your boards are online, the menu will show the name you gave to the board in the line WawiSrv.begin("name") of your sketch. In the figure below, I connected an additional Arduino Uno to my USB which was given the name "My Arduino Uno" whereas my Mega2560 has the name "My Arduino".

67			LUNIT	~ 1	
<u>c</u>		Available interfaces		>	ser1/My Arduino/COM33
00'00		Display formats		>	ser2/My Arduino Uno/COM8
??	$\checkmark$	Online			moChar=0x0238 [1 byte] x 1 VAR_READIN
??		Offline			moChar=0x0238 [1 byte] x 1 VAR_READIN
-MT (		Write row	Ctrl+W	,	moChar=0x0238 [1 byte] x 1 VAR_READIN



## 3.3 The column "Arduino variable name"

In the column "Arduino variable name", you enter the name of the variable you are interested in. The names are case sensitive. WawiLib supports arrays and struct fields. You can use [] to address an individual element in an array and [X..Y] to address a range of elements. If the requested range is too large for the communication protocol maximum message size, use a smaller range. The limitations are on the range requested and not on the size of the array as it is defined in your sketch. So *demoVar* [1000..1010] could be OK whereas *demoVar*[0..1000] is not.

# 3.4 The columns "Actual value" and "Format"

In the column 'Actual value', the actual value of the variable is displayed in accordance with the requested format in the 'Format' column. Multiple elements of an array will be separated by a ';'.

1	Cut Paste	Offline S	etup()	.oop() Write all		
•	Actual value	Write value	Format		v	ariable address and
	0x43		HEX	@demoChar=0x0138 [1	byte] x 1 V	AR_READING_OK -
	67		INT	@demoChar=0x0138 [1	byte] x 1 V	AR_READING_OK -
	67		UINT	@demoChar=0x0138 [1	l byte] x 1 V	AR_READING_OK -
	C		CHAR	@demoChar=0x0138 [1	l byte] x 1 V	AR_READING_OK -
	0b0100'0011		BIT	@domoChar=0v0120 [1	butaly 1 V	AR_READING_OK -
	??		FL	Available interfaces	>	R_READING_OK -
_	??		DO	Display formats	>	▶ HEX
	STR FMT ERR		STI	0.00		TVR
				Offline		UINT
				Setup()		CHAR
	Node		$\checkmark$	Loop()		CHAR
				Write row	Ctrl+W	BII
				Write selected fields		FLOAT
5	active ser	=My Arduino	=CC	Write all	A I+ . 14/	DOUBLE
			-	write all	AIL+VV	STRING
				Cut	Ctrl+X	
				Сору	Ctrl+C	
				Paste	Ctrl+V	
				(lasses ())		-
				Clear row(s)		
				Insert row		
				Insert rows		
				Delete row(s)		
				Clear entire table		
				Variable properties		
				Reset View		

Fig 3.5. WawiLib variable display format selection.

Values can be displayed in various formats:

- bit (BIT), e.g. 0b010'000
- character format (CHAR), e.g. H, \STX, \045
- hexadecimal format (HEX), e.g. 0x48
- signed or unsigned int format(INT/UINT), e.g. 101
- string format (STRING), e.g. Hello world.
- floating point (FLOAT/ DOUBLE) e.g. 3.5e-5

Arrays need to be 0-terminated in order to display them as a string. The index range selected [x..y] needs to include the location of the 0 end-of-string terminator. If not, WawiLib will indicate "STR FMT ERR" (string format error) for the actual value of the variable when displayed as string.

You do not need to select a format for each variable individually, you can select multiple rows and then click right to define the format for all variables. The same goes for other settings (interface types, recorders etc.).

# 3.5 The column "Write value"

In the write value column, you enter a new target value for a variable.

- Enter the value 0x12 as new value for *demoCharAr[0]* as indicated in the next figure.

2 New	Dpen Sa	e ve	🖲 Print	Сору	Ж Cut	Paste	Offlin	e Setup()	Loop	0 Write all	► Continue	e brkpt	⊠ brkpt		
	🗄 demoBool		^		Inter	face/Ard	. ID	/ariable nar	ne	Actual value	Write	value	Format		Variable
	demoChar		_	1	ser1/	My Ardu	ino	demoChar		0x43	0x	12	HEX	@demoChar=	0x0238 [1]
	# demolochar			2		ser1		demoChar		67			INT	@demoChar=	=0x0238 [1
	H demol lint			3		ser1		demoChar		67			UINT	@demoChar=	0x0238 [1 ]
	demoLona			4		ser1		demoChar		С			CHAR	@demoChar=	0x0238 [1 ]
	demoULong			5		ser1		demoChar		0b0100'0011			BIT	@demoChar=	0x0238 [1 ]
	🕀 demoFloat			6		ser1		demoChar		??			FLOAT	@demoChar=	0x0238 [1 ]
	demoDouble			7		ser1		demoChar		??			DOUBLE	@demoChar=	0x0238 [1
	Array			8		ser1		demoChar		- STR FMT ERR	(		STRING	@demoChar=	0x0238 [1
	🗄 demoBoolAr[(	09]		9											
	🗄 demoCharAr[(	024]	× <												>
Index	Time			Node		N	lessage								^
000	29/07/2021 10:	19:16.4	47	ser1/COM	22/My A	rduino C	losing co	mmunication	s port						
001	29/07/2021 10:	19:17.00	05	ser1/COM	22/My A	rduino C	pening s	erial port [CC	M22]	OK					
002	29/07/2021 10:	19:17.00	05	ser1/COM	22/My A	rduino S	etting po	rt parameter	s [baud	=115200 parit	y=N data=	=8 stop=	=1] OK		
003	29/07/2021 10:	19:18.0	78	ser1/COM	22/My A	rduino R	eading se	ttings from	Arduine	<b>D</b> :					
004	20/07/2021 10.	10.10 1	00		22 / A A.	rduina A	rduina lik		02.01	Arduina huffa	: (DV 1		CA hitar	~~~~i~~i~~ti~~	link nod

Fig 3.6. Enter 0x12 as new value for demoChar on line 1.

Write jobs can be triggered in multiple ways:

- Using the toolbar above the grid, press the icon "Write All".
- Using right click on the grid and selecting "Write row" or "Write All".
- Using the shortcuts CTRL+W (Write row) or ALT+W (Write all).
- Pressing the "Enter" key while selecting the "Write value" field when the mode "Autowrite" is activated.

🕙 New	Den 🖻	Save	😕 Print	Сору	<mark>Ж</mark> Cut	Deste Paste	Offline	Setup()	Loop()	Write all		▶ ntinue	e brkpt	⊠ brkpt			
	demoBool		^		Inter	face/Ard	ID Va	riable nam	e	Actual valu	e	Write	value	Format			Variable
		ar.		1	ser1,	My Ardui	no	demoChar		0x43		0x	12	HFX	@dem	oChar=0x	0238 [1
	demolnt			2		ser1		demoChar		67			Availa	able interfa	ces	>	238 [1
	# demoUlnt			3		ser1		demoChar		67			Displa	ay formats		>	238 [1
	demoLona			4		ser1		demoChar		С			Offlin				238 [1
	H demoULon	a		5		ser1		demoChar		0b0100'001	1		C	e 0			238 [1
	H demoFloat	5		6		ser1		demoChar		??			Setup	0			238 [1
	demoDout	ole		7		ser1	1	demoChar		??		~	Loop	0			238 [1
	Array			8		ser1		demoChar	)	STR FMT ER	R		Write	row		Ctrl+W	238 [1
	demoBool/	Ar[09]		9									Write	selected f	स्रोds		
	🖶 demoChar/	Ar[024]	~ <										Write	all		Alt+W	>
Index	Time			Node		N	lessage						Cut			Ctrl+X	^
000	29/07/2021	10:19:16.4	447	ser1/COM	22/My A	rduino C	losing com	munication	s port				Copy			Ctrl+C	
001	29/07/2021	10:19:17.0	005	ser1/COM	22/My A	rduino O	pening ser	al port [CO	M22]	. OK			Paste			Ctrl+V	
002	29/07/2021	10:19:17.0	005	ser1/COM	22/My A	rduino S	etting port	parameters	[baud=	115200 par	ity=N	da l	Character	()			
003	29/07/2021	10:19:18.0	078	ser1/COM22/My Arduino Reading settings from Arduino:								Clear	row(s)				
004	20/07/2021	10.10.10	100	cort/COM	D 2 / N A A	rduina A	rduina libr		02.01 4	rduine huff	ar ai <del>a</del>	~• /D	Insert	row			
×												200	Insert	rows			-
Write va	ariable in curre	ent row to	A LOO	D() Autow	rite on	No reco	rders active	e ser1=N	ly Ardu	ino=COM22	2/115	200,	Delete	e row(s)			2/3/
													Clear	entire table	е		
													Variat	ole propert	ies		
													Reset	View			

Fig 3.7. Trigger write row to write to set 0x12 as new value for *demoChar*.

You can see if WawiLib was able to write by checking the test in the output window. All write jobs will result in a status message in the output window if the option "Display diagnostics messages" is enabled. (Click right in the output window and enable this option in the pop-up menu if required.)

"Autowrite" can be enabled by using the "WawiLib user Preferences and license" dialog box. This dialog box is available under the menu "Settings/Preferences and license".

/awilib u	ser prefer	ences and license	details			×
Layout	License	Startup and exit	Comm. timing	Arduino lib location	Window refresh	
Visible	e columns	-				
🗹 In	terface					
🗹 Va	ariable nar	ne				
🗹 A	ctual valu	e				
<b>₩</b>	/rite value					
🗹 Fo	ormat					
Da	ata record	er				
🗹 A	ddress and	d detalis				
🗌 Dei	tailed prot tailed data	tocol tracing activ a recording tracing	e Jactive			
Au	tomatic s	croll to last line in	output window			
🗌 "Er	nter" key i	n "write value" col	umn field trigger	s write of single variabl	le	
<mark>⊘ Sh</mark>	ow outpu	t (diagnostics) pai	ne			
✓ Sto	ore windo	w size and positior	on exit			
🗹 As	k for conf	irmation to overwi	ite existing file			
Ab	ort comm	nunication on com	munication fault	(time out or fail to op	en comm. port)	
		ОК	Reset se	ettings Cancel		

Fig. 3.8. How to enable "Autowrite".

If you check *"Enter" key in "write value" column field triggers write of single variable,* "Autowrite" will be enabled.

Write values have the same value format as their read counterparts. Format checks are done before writing. If the format of the entered value is not ok, a status message in the output window will indicate the nature of the problem.

#### 3.5.1 Write bit format (BIT)

The figure below illustrates the use of BIT format. Some variables are more than 1 byte long. Writing multiple bytes is done leaving a space between the 8-bit values. The 8-bit values can be preceded by '0b' but this is not mandatory. A single quote (') can be used to separate between the nibbles (nibble = series of 4 bits) to make values more readable.

If the write value is incomplete, WawiLib will try to complete the value; the input value "1 1" will be translated to 0b0000'0001 0b0000'0001 as will "0'1 0'1" but the input value "01" will be flagged illegal writing a 2-byte variable in the Arduino.

In order to get confirmation of your write operations in the output window, you need to enable the option "Display diagnostics messages" by clicking right in the output window (bottom window).

실 Wa	awiLib-PC [C	\Users\Jo	hi\Doc	ument	s\NoN	ame.Wif]-[	C:\Users	s\Johi\D	ocum	ents\NoN	ame.	.Wvf*]						×
File Ec	dit Settings	Help																
1	B		8		1	ж	6		1	T		6	4	Þ	•	Ø		
New	Open	Save	Prin	t 👘	Copy	Cut	Paste	e 0	ffline	Setup()	Lo	op()	Write all	Continue	brkpt	brkpt		
	demoBoo	ol ar	^			Inter	face/Ar	d. ID	Va	riable nar	ne		Actual v	alue	Write	value	Format	^
		nar			1	ser1,	My Ard	uino		demoInt			0x001	2	0x1	2	HEX	@
	# demoint	lai			2	ser1	My Ard	uino		demoInt		0b00	0000'0000	0001'0010	0b01	010	BIT	@
	H demol lin	t			3	ser1,	My Ard	uino		demoInt		0b00	0000'0000	0001'0010	11	0	BIT	@
	H demol or	a			4	ser1,	My Ard	uino		demoInt		0b00	0000'0000	0001'0010	0000000	0000'001	BIT	@
	# demol/l c	ona			5													~
			~	<														>
Index	Time			No	ode			Messag	ge									^
003	29/07/202	1 10:19:18	8.078	se	r1/CON	M22/My A	rduino	Reading	g setti	ngs from	Ardu	lino:						~
<																		>
		Loop	0 A	utowri	te on	No reco	rders ac	tive	s <mark>er1=</mark>	My Arduin	o=C	OM22	2/115200,8	3, <mark>N,1,AVR [</mark>	ITF_LOOP]	msg.ok/t	ot: 3178/3	178

Fig. 3.9. Bit write formatting.

#### 3.5.2 Write hexadecimal format (HEX)

In the figure below (see tracing output window), the fact that 0x was forgotten when writing the value hex 12 is not a problem as WawiLib knows that the format of the output field is HEX. However, writing the value of Z as a hexadecimal value is illegal.

Mew	Dpen	<b>₽</b> Save	Print	Сору	Ж Cut	Paste	Offlir	e Setup()	Loc	ā op()	₩rite all	► Continue	• brkpt	⊠ brkpt		
Avail	able variable	es	^		Inter	face/Ard	. ID	/ariable na	me		Actual va	lue	Writ	e value	Format	
= ser	1\My Ardui	no		1	ser1,	My Ardui	ino	demoInt			0x0102	2	(	)x01	HEX	@demol
		JI JI		2	ser1,	My Ardui	ino	demoInt			0x0102	2	0>	0102	HEX	@demo
	demoCha	r		3	ser1,	My Ardui	ino	demoFloa	t		0x010203	304	0x01	020304	HEX	@demol
	demoUC	nar		4	ser1,	My Ardui	ino	demoDoub	le		1078529	706	0x11	101456		@demol
	demoInt			5	ser1,	My Ardui	ino d	emoDouble	Ar[0]		0x00ABC	DEF	0xA	BCDEF	HEX	@demo
	demoUln	t		6	ser1,	My Ardui	ino d	emoDouble	Ar[2]		0x00000	)12		12	HEX	@demo
	demoLon	g		7	ser1,	My Ardui	ino d	emoDouble	\r[3]		0x000000	000		Z	HEX	@demo
	demoULo	ong	~	¢												>
dex	Time			Node		N	lessage									
00	29/07/202	10:49:46.	153	ser1/COM	22/My A	rduino [F	Row 1] W	riting variab	le dem	noInt	=0x0001 (H	EX) comp	leted.			
01	29/07/202*	10:49:46.	519	ser1/COM	22/My A	rduino [F	Row 2] W	riting variab	e dem	noInt	=0x0102 (H	IEX) comp	leted.			
02	29/07/202	10:49:46.	700	ser1/COM	22/My A	rduino [F	Row 3] W	riting variab	e dem	noFlo	at=0x0102	0304 (HEX	) complete	ed.		
03	29/07/202	10:49:46.	852	ser1/COM	22/My A	rduino [F	Row 4] Ille	egal input va	lue '0	x111(	01456' for '	demoDou	ble' (bad v	/alue/bad va	lue count)	
)4	29/07/2021	10:49:47.	091	ser1/COM	22/My A	rduino [F	Row 5] W	riting variab	e dem	noDo	ubleAr[0]=	0x00ABCD	EF (HEX) o	ompleted.		
)5	29/07/202	10:49:47.	455	ser1/COM	22/My A	rduino [F	Row 6] W	riting variab	e dem	noDo	ubleAr[2]=	0x0000001	12 (HEX) c	ompleted.		
6	29/07/202	10:49:47.	817	ser1/COM	22/My A	rduino [F	Row 71 Ille	gal input va	lue 'z'	for '	demoDout	leAr[3]' (ba	ad value/h	ad value co	(int)	

Fig. 3.10. Bit Hex formatting.

# 3.5.3 Write integer format (INT/UINT)

Writing of (signed) integers (INT) and unsigned integers (UINT) is straightforward and does not need further detailing.

실 Wa	wiLib-PC [C:	Users\Joł	<mark>hi∖Docu</mark>	ments\NoN	ame.Wif]-[0	C:\Users\Joh	ni\Docun	nents\NoNa	ame. <mark>Wv</mark> f*	]				-		×
Mew New	Den Open	5ave	🖨 Print	Copy	لا Cut	🖻 Paste	The offline	Setup()	Toop()	Write	<b>e all</b> Continue	e brkpt	⊠ brkpt			
🖻 Avai	able variabl	es	^		Inter	ace/Ard. II	D Vari	able name	Actual	value	Write value	Format	1	Variak	le addre	ss and sta
l se	r1\My Ardu	ino		1	ser1/	My Arduino	de	emoBool	-5		-5	INT	@demoBool=0x0	0239 [	1 byte] x	1 VAR_R
		bie		2	ser1/	My Arduino	de	emoChar	2		2	INT	@demoChar=0x0	0238 [	1 byte] x	1 VAR_R
	H demoCha	ar .		3	ser1/	My Arduino	de	moUChar	46	5	-300	INT	@demoUChar=0	x0237	[1 byte]	x 1 VAR
		har		4	ser1/	My Arduino		demoInt	4		4	INT	@demoInt=0x02	35 [2	byte] x 1	VAR REA
	e demoloci			5	ser1/	My Arduino	d	emoUInt	5		5	INT	@demoUInt=0x0	233 [2	byte] x	1 VAR_R
	demoUln	t		6	ser1/	My Arduino	de	emoLong	50000	0000	50000000	INT	@demoLong=0x	022F [	4 byte] x	1 VAR_R
	demoLor	a		7	ser1/	My Arduino	de	demoULong		883		INT	@demoULong=0	x022E	[4 byte]	x 1 VAR
	🗄 demoULo	ong		8	ser1/	My Arduino	de	demoFloat		78		INT	@demoFloat=0x	0227 [	4 byte] x	1 VAR_R
	H demoFlo	at		9	ser1/	My Arduino	de	emoFloat	1.0930	1e-43		float	@demoFloat=0x	0227 [	4 byte] x	1 VAR_R
	🗄 demoDo	uble		10	ser1/	ser1/My Arduino		demoFloat		0004E		HEX	@demoFloat=0x	0227 [	4 byte] x	1 VAR_R
	Array			11	ser1/	My Arduino	der	noDouble	56	5	56	INT	@demoDouble=	0x022	3 [4 byte	] x 1 VAF
	demoBoo	olAr[09]	~	<									•			>
Index	Time			Node		Mes	sage									^
004	29/07/202	1 10:49:47	.091	ser1/COM	M22/My Ar	duino [Rov	v 5] Writ	ing variable	demoDo	ubleA	r[0]=0x00ABC[	DEF (HEX)	completed.			
005	29/07/202	1 10:49:47	.455	ser1/COM	M22/My Ar	duino [Rov	w 6] Writ	ing variable	demoDo	ubleA	r[2]=0x000000	12 (HEX)	completed.			
006	29/07/202	1 10:49:47	.817	ser1/COM	M22/My Ar	duino [Rov	v 7] Illeg	al input valu	ue 'z' for '	demo	DoubleAr[3]' (b	ad value/	bad value count)			
007	29/07/202	1 10:53:35	.324	ser1/COM	M22/My Ar	duino [Rov	v 1] Writ	ing variable	demoBo	ol=-5 (	(INT) complete	d.				
800	29/07/202	1 10:53:37	.652	ser1/COM	M22/My Ar	duino [Rov	v 2] Writ	ing variable	demoCh	ar=2 (I	NT) completed	1.				
009	29/07/202	1 10:53:39.	.596	ser1/COM	M22/My Ar	duino [Rov	w 3] Out	of range va	lue '-300	for 'de	emoUChar'					
010	29/07/202	1 10:53:40	.591	ser1/COM	M22/My Ar	duino [Rov	v 4] Writ	ing variable	demoInt	=4 (IN	T) completed.					
011	29/07/202	1 10:53:41	.015	ser1/COM	M22/My Ar	duino [Rov	v 5] Writ	ing variable	demoUl	nt=5 (II	NT) completed	i				
012	29/07/202	1 10:53:57	.116	ser1/COM	M22/My Ar	duino [Rov	v 6] Out	of range va	lue '5000	00000	000' for 'demo	Long'				
013	29/07/202	1 10:54:07	.704	ser1/COM	M22/My Ar	duino [Rov	v 6] Writ	ing variable	demoLo	ng=50	000000 (INT) c	ompleted	1.			
<	~~~~~					1.1	~~ · · · ·				• • •					>
				Loop()	Autowrite	on Nor	ecorders	s active s	er1=My	Arduin	o=COM22/115	5200,8,N,	I,AVR [ITF_LOOP]	msg.o	k/tot: 44	83/4483

Fig. 3.11. Integer formatting.

#### Note:

In the table above you see *demoFloat* represented as an INT and written with the value 78. On line 9 you can see the result if the 4-byte value is interpreted as a mantisse + exp value, that is the internal representation of a float. You get a floating point value of 1.09e-43!

If you tell WawiLib to visualise the variable *demoFloat*, it means that WawiLib has to take this 4 byte memory location an see it as a type indicated in the "format column" regardeless of what you have defined in your Sketch.

## 3.5.4 Write floating point format (FLOAT / DOUBLE)

WawiLib supports different formats of floating-point variables. You can use scientific notation (1.25e-1) or standard notation (0.125).

Depending on the platform, the size of a double variable in the Arduino can be 4-byte (MEGA) or 8byte (DUE). WawiLib is able to differentiate between the 2 floating point variable types. Therefore, if you connect a MEGA2560 to WawiLib, double and float formatting are identical (4 byte), but if you

没 Wa	wiLib-PC [C:\	Users\Jol	hi∖Docu	ments\NoNa	me.Wif]-[	C:\Users\.	Johi\Docu	ments\NoNa	ame.Wvf*	]		_		$\times$
File Ed	dit Settings	Help												
1	B		8	<b>1</b>	X	<u> </u>	1	<u>T</u>	<b>T</b>	Ł	• •	٠	Ø	
New	Open	Save	Print	Сору	Cut	Paste	Offline	Setup()	Loop()	Writ	e all Continue	e brkpt	brkpt	
Avai	lable variable	s	^		Inter	face/Ard	. ID Vai	iable name	Actual	value	Write value	Format		V
	Single variab			1	ser1,	/My Ardui	ino d	emoFloat	0.0	)1	3.14	FLOAT	@demoFlo	oat=0x0
Ī		l		2	ser1,	/My Ardui	ino d	emoFloat	0.0	1	1E-10	FLOAT	@demoFlo	oat=0x0
	demoChai	1		3	ser1,	/My Ardui	ino d	emoFloat	0.0	1	0.1e-1	FLOAT	@demoFla	oat=0x0
	H demoUCh	ar		4	ser1,	/My Ardui	ino de	moDouble	6.89	59	-1.2e-8	DOUBLE	@demoDo	ouble=0
	demoInt			5	ser1,	/My Ardui	ino de	moDouble	6.89	59	-5.6	DOUBLE	@demoDo	ouble=0
	demoUInt			6	ser1,	/My Ardui	ino de	moDouble	6.89	59	6.8959	DOUBLE	@demoDo	ouble=0
	H demoLong	9												
	H domol II o	na	~	<										>
Index	Time			Node		N	lessage							
000	29/07/2021	11:02:58	.392	ser1/COM	22/My A	rduino [F	Row 1] Illeg	gal input val	ue '3.14' f	or 'der	moFloat' (bad	value/bac	value cour	nt)
001	29/07/2021	11:03:03	.417	ser1/COM	22/My A	rduino [F	Row 1] Illeg	gal input val	ue '3,14' f	or 'der	moFloat' (bad	value/bac	value cour	nt)
002	29/07/2021	11:03:07	.292	ser1/COM	22/My A	rduino [F	Row 1] Illeg	gal input val	ue '3.14' f	or 'der	moFloat' (bad	value/bac	value cour	nt)
003	29/07/2021	11:03:28	.222	ser1/COM	22/My A	rduino [F	Row 1] Wri	ting variable	demoFlo	oat=3.1	14 (FLOAT) com	pleted.		
004	29/07/2021	11:03:31	.335	ser1/COM	22/My A	rduino [F	Row 2] Wri	ting variable	demoFlo	oat=1e	-10 (FLOAT) co	mpleted.		
005	29/07/2021	11:03:35	.504	ser1/COM	22/My A	rduino [F	Row 3] Wri	ting variable	demoFlo	oat=0.0	01 (FLOAT) con	pleted.		
006	29/07/2021	11:03:38	.102	ser1/COM	22/My A	rduino [F	Row 4] Wri	ting variable	demoDo	ouble=	-1.2e-08 (DOU	BLE) com	pleted.	
007	29/07/2021	11:03:41	.155	ser1/COM	22/My A	rduino [F	Row 5] Wri	ting variable	demoDo	ouble=	-5.6 (DOUBLE)	complete	d.	
800	29/07/2021	11:03:44	.236	ser1/COM	22/My A	rduino [F	Row 6] Wri	ting variable	demoDo	ouble=	6.8959 (DOUB	LE) comp	eted.	
<														>
	Loop	0 Auto	owrite o	n No reco	ders act	ive ser	1=My Ard	uino=COM2	2/115200	),8,N,1	AVR [ITF_LOOI	P] msg.ok	/tot: 5247/	5247

# Fig. 3.12. Floating point formatting.

# 3.5.5 Write character format (CHAR)

邊 Wa	awiLib-PC [C	:\Users\Jol	hi∖Docı	uments\NoN	ame.Wif]-	[C:\Users\Jo	hi\Docu	ments\NoNa	ame.Wvf*	]			_		×
File Ed	dit Settings	Help													
2	D		8	1	ж	ß	1	10	1	4	•	٠	Ø		
New	Open	Save	Print	Сору	Cut	Paste	Offline	e Setup()	Loop()	Write	e all Continue	e brkpt	brkpt		
Avai	lable variabl	es	^		Inter	face/Ard.	ID Va	riable name	Actual	value	Write value	Format		Vari	ia <mark>ble ad</mark> o
	Single varia	hle		1	ser1	/My Arduin	0 0	lemoChar	A	()	A	CHAR	@demoChar	=0x023	3 [1 byte]
1				2	ser1	/My Arduin	0	demoInt	A	В	AB	CHAR	@demoInt=0	)x0235	[2 byte] x
	# demoCh	ar		3	ser1	/My Arduin	0 0	lemoFloat	ABO	CD	ABCD	CHAR	@demoFloat	=0x022	7 [4 byte
	# demoUC	har		4	ser1	/My Arduin	o de	moDouble	AB	CE	ABCE	CHAR	@demoDoul	ble=0x0	223 [4 by
	# demoint											-			
	demoUln	t													
	H demoLor	ng													
	damalil		~	<											>
Index	Time			Node		Me	essage								
000	29/07/202	1 11:05:59	.554	ser1/COM	122/My A	rduino [Ro	w 1] Wri	ting variable	demoCh	nar=A (0	CHAR) comple	ted.			
001	29/07/202	1 11:05:59	.945	ser1/COM	/22/My A	rduino [Ro	w 2] Wri	ting variable	demoln	t=AB (C	HAR) complet	ted.			
002	29/07/202	1 11:06:00	.157	ser1/COM	122/My A	rduino [Ro	w 3] Wri	ting variable	demoFle	oat=AB	CD (CHAR) co	mpleted.			
003	29/07/202	1 11:06:00	.340	ser1/COM	122/My A	rduino [Ro	w 4] Wri	ting variable	demoDo	ouble=/	ABCE (CHAR) o	ompleted	d.		
													_		
<															>
		Loc	pp()	Autowrite or	No re	corders acti	ive se	r1=My Ardui	ino=CON	/22/11	5200,8,N,1,AV	R [ITF_LO	OP] msg.ok/to	ot: 5476,	/5476

# Fig. 3.12. Char formatting.

Characters input supports multiple input formats. Variables larger than 1 character can be written. The number of characters must match the length of the variable. You can see the size of the variable in the last column of the grid. Writing to arrays of non CHAR data types is supported.

WawiLib supports character input in different formats:

- Literal value, e.g. ABCD.
- ASCI code for all digits, e.g. \065,\066,\067,\068 (=ABCD).
- Code for special characters in the range 0..31, e.g. \ETX,\NUL (capital case, see ASCII table below).
- C Escape sequences, e.g. \r,\n,\t (lower case).

Some values of the ASCII table cannot be displayed as a letter, because they have a special function. WawiLib character output has the following precedence for displaying character values:

- Literal value, e.g. A.
- C Escape sequences, e.g. \r.
- Operand code for special characters in the range 0..31, e.g. \STX\ETX\NUL (! capital case).
- WawiLib ASCI code for all digits, e.g. \065.

	ait Settings	нер	-			-	and the	-	and some						
2			8		all a	<b>D</b>			Ĩ.	0 M 1	•		Ø		
New	Open	Save	Prin	с Сору	Cut	Paste	Offline	Setup()	Loop	0 Write all	Continue	brkpt	brkpt		
Avai	lable variable	es	^		Inter	face/Ard.	DV	ariable na	me	Actual value	Write	e value	Format		
le se	Cingle uppiel	no		1		ser1	de	moCharAr[	03]	A\2403D	A	BCD	CHAR	@demoCharA	r=(
				2		ser1	de	moCharAr[	47]	ABCD	\065\06	6\067\068	CHAR	@demoCharA	r=(
	H demoCha	ar		3		ser1	der	noCharAr[8	311]	\STX\f\n\NUL	\STX\F	F\LF\NUL	CHAR	@demoCharA	r=(
	H demoUC	har		4		ser1	dem	oCharAr[1	213]	\r\n	١	r\n	CHAR	@demoCharA	.r=(
	demoint			5		ser1	d	emoCharA	r[0]	A	١	DEL	CHAR	@demoCharA	r=(
	🗄 demoUln	t		6		ser1	d	emoCharA	r[1]	\240	1	240	CHAR	@demoCharA	r=(
	H demoLor	g		7		ser1	d	emoCharA	r[2]	3		3	CHAR	@demoCharA	r=C
	🗄 demoULo	ng		8		ser1		demoInt		A\t	1	4\t	CHAR	@demoInt=0	<mark>(</mark> 02
	🗄 demoFloa	at		9											
	🗄 demoDo	uble		10											
ė	Array			11											
	🗄 demoBoo	olAr[09]		12											
	🗄 demoCha	rAr[024]													
	🕀 demol IC	harAr(0.91	~	<											>
Index	Time			Node		Me	ssage								
000	29/07/202	111:11:19.	765	ser1/CON	122/My A	rduino [Ro	w 1] Writ	ng variable	e demo	CharAr[03]=A	;B;C;D (CH	IAR) comp	leted.		
001	29/07/202	111:11:19.	917	ser1/CON	122/My A	rduino [Ro	w 2] Writ	ng variable	e demo	CharAr[47]=A	;B;C;D (CH	IAR) comp	leted.		
002	29/07/202	11:11:20.	099	ser1/CON	122/My A	rduino [Ro	w 3] Writ	ng variable	e demo	CharAr[811]=	\STX;\f;\n	NUL (CHA	R) compl	eted.	
003	29/07/202	111:11:20.	280	ser1/CON	122/My A	rduino [Ro	w 4] Writ	ng variable	e demo	CharAr[1213]	=\r;\n (CH	AR) comple	eted.		
004	29/07/202	111:11:20.4	433	ser1/CON	122/My A	rduino [Ro	w 5] Illeg	al input ' \[	DEL' for	'demoCharAr[	0]' (char)				
005	29/07/202	11:11:20.	644	ser1/CON	122/My A	rduino [Ro	w 6] Writ	ng variable	e demo	CharAr[1]=\24	) (CHAR)	completed.			
006	29/07/202	111:11:20.	824	ser1/CON	122/My A	rduino [Ro	w 7] Writ	ng variable	e demo	CharAr[2]=3 (C	HAR) con	npleted.			
007	29/07/202	111:11:21.	275	ser1/CON	122/My A	rduino [Ro	w 8] Writ	ng variable	e demo	Int=A\t (CHAR	) complet	ed.			
<															>

Fig. 3.12. Char formatting of arrays.



Fig. 3.13. Ascii table.

These values are represented in the format  $\ldots$  where  $\ldots$  is an abbreviation of the function (example STX). Alternatively can be used ddd for values that have no symbol nor a special function (typically values above 127 (example 145 or 064):

#### 3.5.6 Write string format (STRING)

Mew	Dpen		Prin	t Copy	۸ Cut	Paste	Offline	Setup()	Loop	o() Write al	► Continue	e brkpt	⊠ brkr	ot	
	demoLon	gAr[09]	^		Inter	face/Ard.	ID V	ariable na	me	Actual valu	e W	rite value		Format	:
	demoULc	ongAr[09]		1	ser1/	My Arduir	no der	noCharArl	0 121	Hello work	I H	ello world		STRING	@de
	demoFloa	atAr[09]		2	ser1/	My Arduir		demoDout	ble					STRING	@da
	demoDou	ubleAr[09]		3	ser1/	My Arduir	no dem	oDoubleA	Ar[0.2]	163	\STX\FT	X\Svlvester	NUI	STRING	@de
	-[4 byte	J X 10		4	ser1/	My Arduir	no dem	oDoubleA	Ar[0, 2]	163	(5171(21)	163		STRING	@de
	- demoL	OubleAr[U]		5	ser1/	My Arduir	no dem	noDoubleA	Ar[3.,7]	Hello world	Hell	o world\r\r	n	STRING	@de
	- demoL	oubleAr[1]		6		,									Gui
	- demo[	OubleAr[2]													
	- demo[	oubleAr[4]													
	-demo[	oubleAr[5]													
	demoD	OoubleAr[6]													
	- demoD - demoD	oubleAr[6]													
	-demoD -demoD -demoD	)oubleAr[6] )oubleAr[7] )oubleAr[8]													
	- demoD - demoD - demoD - demoD	)oubleAr[6] )oubleAr[7] )oubleAr[8] )oubleAr[9]													
	- demoD - demoD - demoD - demoD	)oubleAr[6] )oubleAr[7] )oubleAr[8] )oubleAr[9]	>	<						_					>
Index	- demoD - demoD - demoD - demoD	oubleAr[6] oubleAr[7] oubleAr[8] oubleAr[9]	<	< Node		Messa	ge								>
Index 000	-demoE -demoE -demoE -demoE Time 29/07/2021	)oubleAr[6] )oubleAr[7] )oubleAr[8] )oubleAr[9]	<b>~</b> 32	< Node ser1/COM22/	My Arduir	Messa Ino [Row 1	ge ] Writing v	variable de	moCha	ırAr[012]=H€	llo world (S	TRING) cor	mplete	ed.	>
Index 000 001	- demoE - demoE - demoE - demoE - demoE 29/07/2021 29/07/2021	DoubleAr[6] DoubleAr[7] DoubleAr[8] DoubleAr[9] I 11:14:27.9 I 11:14:29.5	<b>v</b> 32 10	Node ser1/COM22/ ser1/COM22/	My Arduir My Arduir	Messa Io [Row 1 Io [Row 2	ge ] Writing v ?] Writing v	variable de variable de	moCha	ırAr[012]=He uble= (STRIN	ello world (S 5) complete	STRING) cor	mplete	ed.	>
Index 000 001 002	- demoE - demoE - demoE demoE 29/07/2021 29/07/2021 29/07/2021	DoubleAr[6] DoubleAr[7] DoubleAr[8] DoubleAr[9] I 11:14:27.9 I 11:14:29.5 I 11:15:38.5	× 32 10 57	Node ser1/COM22/ ser1/COM22/ ser1/COM22/ ser1/COM22/	My Arduir My Arduir My Arduir	Messa Ino [Row 1 Ino [Row 2 Ino [Row 1	ge ] Writing v ] Writing v ] Writing v	variable de variable de variable de	moCha moDou moCha	ırAr[012]=Ha uble= (STRIN0 ırAr[012]=Ha	ello world (S G) complete ello world (S	STRING) cor ed. STRING) cor	mplete	ed. ed.	>
Index 000 001 002 003	- demoE - demoE - demoE 29/07/2021 29/07/2021 29/07/2021 29/07/2021	DoubleAr[6] DoubleAr[7] DoubleAr[8] DoubleAr[9] I 11:14:27.9 I 11:14:29.5 I 11:15:38.5 I 11:15:39.4	× 32 10 57 09	Node ser1/COM22/ ser1/COM22/ ser1/COM22/ ser1/COM22/	My Arduir My Arduir My Arduir My Arduir	Messa Ioo [Row 1 Ioo [Row 2 Ioo [Row 1 Ioo [Row 2	ge ] Writing v ] Writing v ] Writing v 2] Writing v	variable de variable de variable de variable de	moCha moDou moCha moDou	ırAr[012]=He uble= (STRIN ırAr[012]=He uble= (STRIN	ello world (S G) complete ello world (S G) complete	STRING) cor ed. STRING) cor ed.	mplete	ed. ed.	>
Index 000 001 002 003 004	- demoD - demoD - demoD demoD 29/07/2021 29/07/2021 29/07/2021 29/07/2021	DoubleAr[6] DoubleAr[7] DoubleAr[8] DoubleAr[9] I 11:14:27.9 I 11:14:29.5 I 11:15:38.5 I 11:15:39.4 I 11:15:40.0	× 32 10 57 09 79	Node ser1/COM22/ ser1/COM22/ ser1/COM22/ ser1/COM22/ ser1/COM22/ ser1/COM22/	My Arduir My Arduir My Arduir My Arduir My Arduir	Messa 100 [Row 1 100 [Row 2 100 [Row 2 100 [Row 3 100 [Row 3	ge ] Writing v ] Writing v ] Writing v 2] Writing v 2] Writing v 3] Illegal in	variable de variable de variable de variable de variable de put value "	moCha moDou moCha moDou \STX\ET	ırAr[012]=He uble= (STRIN ırAr[012]=He uble= (STRIN IX\Sylvester\I	ello world (S G) complete ello world (S G) complete NUL' for 'dei	STRING) cor ed. STRING) cor ed. moDouble/	mplete mplete Ar[02	ed. ed. ]' (bad st	>
Index 000 001 002 003 004 005	- demoD - demoD - demoD demoD 29/07/2021 29/07/2021 29/07/2021 29/07/2021 29/07/2021	DoubleAr[6] DoubleAr[7] DoubleAr[8] DoubleAr[9] I 11:14:27.9 I 11:14:29.5 I 11:15:38.5 I 11:15:39.4 I 11:15:40.0 I 11:15:41.3	× 332 510 557 .009 779 447	Node ser1/COM22/ ser1/COM22/ ser1/COM22/ ser1/COM22/ ser1/COM22/ ser1/COM22/ ser1/COM22/	My Arduir My Arduir My Arduir My Arduir My Arduir My Arduir My Arduir	Messa 100 [Row 1 100 [Row 2 100 [Row 2 100 [Row 3 100 [Row 4	ge ] Writing \ ] Writing \ ] Writing \ ] Writing \ ] Illegal in ] Writing \	variable de variable de variable de variable de put value " variable de	moCha moDou moCha moDou \STX\E1	ırAr[012]=Hq uble= (STRIN ırAr[012]=Hq uble= (STRIN IX\Sylvester\I ubleAr[02]=1	ello world (S 5) complete ello world (S 5) complete NUL' for 'dee 63 (STRING	STRING) cor ed. STRING) cor ed. moDouble, 5) complete	mplete mplete Ar[02	ed. ed. ]' (bad st	> tring v
Index 000 001 002 003 004 005 006	- demo - demo - demo demo 29/07/2021 29/07/2021 29/07/2021 29/07/2021 29/07/2021 29/07/2021 29/07/2021	oubleAr[6] oubleAr[7] oubleAr[7] oubleAr[8] oubleAr[9] 111:14:27.9 111:14:29.5 111:15:38.5 111:15:39.4 111:15:40.0 111:15:41.3 111:15:42.7	<ul> <li>32</li> <li>10</li> <li>57</li> <li>09</li> <li>79</li> <li>447</li> <li>06</li> </ul>	CNODE Ser1/COM22/ Ser1/COM22/ Ser1/COM22/ Ser1/COM22/ Ser1/COM22/ Ser1/COM22/ Ser1/COM22/	My Arduir My Arduir My Arduir My Arduir My Arduir My Arduir My Arduir	Messa o (Row 1 o (Row 2 o (Row 1 o (Row 2 o (Row 3 o (Row 4 o (Row 4	ge ] Writing \ ] Writing \ ] Writing \ ] Writing \ ] Illegal in I] Writing \ i] Writing \ i] Writing \	variable de variable de variable de variable de put value " variable de variable de	moCha moDou moCha moDou \STX\E1 moDou moDou	ırAr[012]=Hı ıble= (STRINı ırAr[012]=Hı ıble= (STRINı IX\Sylvester\/ ıbleAr[02]=' ıbleAr[37]=ł	ello world (S 5) complete ello world (S 5) complete NUL' for 'der 63 (STRING fello world	GTRING) cor ed. STRING) cor ed. moDouble. 5) complete c. (STRING) d	mplete mplete Ar[02 ed. compl	ed. ed. ']' (bad st eted.	> tring v
Index 000 001 002 003 004 005 006	- demoD - demoD - demoD 29/07/2021 29/07/2021 29/07/2021 29/07/2021 29/07/2021 29/07/2021 29/07/2021	oubleAr[6] oubleAr[7] oubleAr[7] oubleAr[8] oubleAr[9] 11:14:27.9 11:14:29.9 11:15:38.5 11:15:38.4 11:15:39.4 11:15:39.4 11:15:40.0 11:15:41.3 11:15:42.7	<ul> <li>32</li> <li>10</li> <li>57</li> <li>09</li> <li>79</li> <li>447</li> <li>06</li> </ul>	Node ser1/COM22/ ser1/COM22/ ser1/COM22/ ser1/COM22/ ser1/COM22/ ser1/COM22/ ser1/COM22/	My Arduir My Arduir My Arduir My Arduir My Arduir My Arduir My Arduir	Messa o [Row 1 o [Row 2 o [Row 2 o [Row 2 o [Row 3 o [Row 4 o [Row 5	ge ] Writing v ] Writing v ] Writing v 2] Writing v 2] Illegal in 1] Writing v 5] Writing v	variable de variable de variable de variable de put value " variable de variable de	moCha moDou moCha moDou \STX\E1 moDou	arAr[012]=H4 uble= (STRIN arAr[012]=H4 uble= (STRIN ubleAr[02]= ubleAr[02]= ubleAr[37]=H	ello world (S 5) complete ello world (S 5) complete NUL' for 'dei 63 (STRING Hello world.	STRING) cor ed. STRING) cor ed. moDouble. 5) complete 6) complete (STRING) (	mplete mplete Ar[02 ed. compl	ed. ed. ]' (bad st eted.	> tring v

Fig. 3.14. Char formatting of strings (zero terminated char arrays).

Strings are typically arrays of characters, they will be treated in the next chapter. However, if you have a 4-byte variable, you can write a 3-character string + tailing 0 to this variable as on line 4. Characters in a string that are non digit will be displayed as '.'. If you want to see all characters in detail, switch to 'char' format.

Available variab ser1\My Ardu Single varia demoCh demoCh demoUl demoUl demoUl demoUl demoUl demoUl demoUl demoUl demoUl	bles Juino iable ool har IChar nt IInt ong ILong		1 2 3 4 5 6 7	Interface/. ser1 ser1 ser1/My A ser1/My A ser1/My A	Ard. ID rduino rduino	Variable na demoBoolA demoCharAr demoIntAr[3 demoUIntAr[ demoFloatA demoDouble	me r[0] [24] 85] 24] r[1]	Actual value 0b0000'0001 Ilo 0;0;0 7;65528;9 6.3	Write value 1 ABC 7;-8;9 4;5;6 6.3	Format bit char int uint float	@demoBoolAr=0 @demoCharAr=0 @demoIntAr=0x @demoUIntAr=0	Variable a 0x0219 [1 0x0200 [1 03F2 [2 by	address a byte] x 10 byte] x 29 /te] x 10 -	nd s ' 0 ` 5 `
Sert VMy Ardd Single varia Game democh demo	iable ool har IChar nt IInt ong ILong		1 2 3 4 5 6 7	ser1 ser1 ser1/My A ser1/My A ser1/My A	rduino rduino	demoBoolA demoCharAr[ demoIntAr[3 demoUIntAr[ demoFloatA demoDouble	r[0] [24] 35] 24] r[1]	0b0000'0001 Ilo 0;0;0 7;65528;9 6.3	1 ABC 7;-8;9 4;5;6 6.3	bit char int uint	@demoBoolAr=0 @demoCharAr=0 @demoIntAr=0x @demoUIntAr=0	0x0219 [1 0x0200 [1 03F2 [2 by	byte] x 10 byte] x 29 yte] x 10 -	0 ` 5 ` VA
Single Vall     demoBo     demoCh     demoCl     demoLo     demoIl     demoLo     demoIl	ool har IChar It IInt ong ILong		2 3 4 5 6 7	ser1 ser1 ser1/My A ser1/My A ser1/My A	rduino rduino	demoCharAr demoIntAr demoUIntAr demoFloatA demoDouble	[24] 35] 24] r[1]	llo 0;0;0 7;65528;9 6.3	ABC 7;-8;9 4;5;6 6.3	char int uint	@demoCharAr=0 @demoIntAr=0x @demoUIntAr=0	0x0200 [1 03F2 [2 by	byte] x 2: vte] x 10 -	5 ` VA
demoCh     demoCh     demoL0	ihar IChar IChar Int Int Ing ILong		3 4 5 6 7	ser1 ser1/My A ser1/My A ser1/My A	rduino rduino	demoIntAr[3 demoUIntAr[ demoFloatA demoDouble	85] 24] .r[1]	0;0;0 7;65528;9 6.3	7;-8;9 4;5;6 6.3	int uint float	@demoIntAr=0x @demoUIntAr=0	03F2 [2 b	yte] x 10 -	VA
demoU(	IChar Int Int Ing ILong		4 5 6 7	ser1/My A ser1/My A ser1/My A	rduino rduino	demoUIntAr[ demoFloatA demoDouble	24] .r[1]	7;65528;9 6.3	4;5;6 6.3	uint float	@demoUIntAr=0			
demote d	nt IInt ong ILong		5 6 7	ser1/My A ser1/My A ser1/My A	rduino rduino	demoFloatA demoDouble	.r[1]	6.3	6.3	float		XUSDE [Z	byte] x 10	) V
demoUl demoLo demoUL demoFlo	llnt ong ILong		6 7	ser1/My A	rduino	demoDouble	demoFloatAr[1]			float double	@demoFloatAr=	0x0366 [4	byte] x 1	0
e demoLo demoUL demoFlo demoFlo	ong ILong		7	ser1/My A		rduino demoDoubleA		3.14152	3.14152		@demoDoubleA	r=0x033E	[4 byte] x	(10
e demoUL demoFlo demoDo	Long			ser1/My Arduino ser1/My Arduino		demoUCharAr[2] demoUCharAr[3]		A	A	CHAR CHAR	R @demoUCharAr=0x0406 [1 b			10
demoFlo			8					В	В		R @demoUCharAr=0x0406 [1 byte]			
r demoDr	loat		9	ser1/My A	rduino	demoUChar/	Ar[4]	С	С	CHAR	@demoUCharAr	=0x0406 [	1 byte] x	10
<	ouble	~	10											
Sector Se	>	<												>
Index Time			Node		Messa	ge								1
000 29/07/202	21 11:14:2	7.932	ser1/COM2	2/My Arduin	Row	] Writing variat	ole de	moCharAr[012	2]=Hello world	(STRING)	completed.			
001 29/07/202	21 11:14:2	9.510	ser1/COM2	2/My Arduin	Row a	2] Writing variat	ole de	moDouble= (S	TRING) comple	eted.				
002 29/07/202	21 11:15:3	8.557	ser1/COM2	2/My Arduin	Row	1] Writing varia	ole de	moCharAr[012	2]=Hello world	(STRING)	completed.			
003 29/07/202	21 11:15:3	9.409	ser1/COM2	2/My Arduin	Row 2	2] Writing varia	ole de	moDouble= (S	TRING) comple	eted.				
004 29/07/202	21 11:15:4	0.079	ser1/COM2	2/My Arduin	Row 3	3] Illegal input v	alue "	STX\ETX\Sylve	ster\NUL' for 'o	demoDou	bleAr[02]' (bad s	string valu	e or too l	ong
005 29/07/202	21 11:15:4	1.347	ser1/COM2	2/My Arduin	Row 4	4] Writing varial	ole de	moDoubleAr[0.	2]=163 (STRI	NG) comp	leted.			

### 3.5.7 Writing arrays of values



Array write values need to be separated in default by a ';' (except for char arrays and strings). On line 2 in the table above, you can see an example of a write to 3 array elements [2,3,4] using a single value ABC in the write value column.

String variables can be written using plain text as you can see in the table below. Do not forget that the receiving array must also be able to contain the 0 character at the end of the string. WawiLib will automatically add the 0 character when writing to a variable represented in string format.

실 Wa	wiLib-PC [C:\Users\Joh	i∖Do	cuments\NoNa	me.Wif]-	C:\Users\Joh	i∖Docum	ents\NoN	ame.Wvf*	]				_		×
File Ec	lit Settings Help Den Save	Pri	nt Copy	<mark>Ж</mark> Cut	Daste	The offline	Setup()	Loop()	₩rite all	► Continue	• brkpt	⊠ brkpt			
	🖶 demoFloat	^		Interf	ace/Ard. ID	Vari	able nam	•	Actual	value	Writ	e value	Format		^
	demoDouble		1		ser1	demo	CharAr[0	[2]	Hello V	Vorld	Hello	o World	string	@demoC	harA
			2		ser1	demo	CharAr[0*	12] H	Hello World	NUL\NUL			char	@demoC	harA
	demoCharAr[0.24]		3	ser1/I	My Arduino	dem	oCharAr[10	0]	d				CHAR	@demoC	harA
	-[1 byte] x 25		4	ser1/I	My Arduino	dem	oCharAr[1	1]	\NI	JL			CHAR	@demoC	harA
	-demoCharAr[0]		5	ser1/I	My Arduino	dem	oCharAr[12	2]	0x0	00			hex	@demoC	harA
	-demoCharAr[1]	6	ser1/I	My Arduino	dem	noCharAr[0	]	72					@demoCl	harA	
	-demoCharAr[2]		7	ser1/i	My Arduino	dem	noCharAr[1	]	10	1			int	@demoC	harA
	-demoCharAr[3]	0	8												、 <sup>×</sup>
		· .													
Index	lime		Node		Message										
012	29/07/2021 11:53:01.	367	ser1/COM22/N	Arduir	no [Row 5]	Writing v	ariable der	noFloatA	r[1]=6.3 (FL	OAI) compl	leted.				
013	29/07/2021 11:53:06.	/65	ser1/COM22/N	Arduir	10 [Row 5] I	llegal inp	out value '4	;5;6' for 'e	demoFloat	Ar[1]' (bad v	alue/bad	value co	unt)		1.0
014	29/07/2021 11:53:13.	525	ser1/COM22/N	Arduir	no [Row 4]	Writing v	ariable der	noUIntAr	[24]=7;-8;	9 () complet	ed.				
015	29/07/2021 11:53:15.	/68	ser1/COM22/N	Arduir	10 [Row 3] I	llegal inp	out value 'A	BC for o	iemoIntAr[	35] (bad va	alue/bad	value cou	unt)		
016	29/07/2021 11:53:18.0	000	ser1/COM22/N	iy Arduir		liegal inp	out value 'A	tor den			e/bad vali	ue count	)		
010	29/07/2021 11:53:32.	410	ser1/COM22/N	Arduir	10 [ROW 5]		ariable der	norioatAl	(01-06000		etea.	d			~
<	23/07/2021 11:33'417	+ 117	SPLT/CON/22/N	ATTIN	IO IROW III	withind V	aname der	10500/Ar	01=00000	DOOUT (BIL)	commen	-(1			>
		Loop	o() Autowrite	on N	o recorders	active	ser1=My	Arduino=	COM22/11	5200,8,N,1,/	AVR [ITF	.00P] m	sg.ok/tot	: 12284/12	284

Fig. 3.16. Writing arrays of values.

# 4 Further reading

This demo demonstrates how to read and write variables with WawiLib. Variables of different sizes can be read and written in various formats. Arrays and parts of arrays can also be read and written. Special attention is required when reading and writing strings as they have to be zero terminated. Char formats are supported in multiple forms.

Recording of variables can be executed "on change", "on timer" or both. Recording can also be done with one file per hour or per day to make the generated files more manageable. WawiLib supports links via the USB programming interface, WiFi, cabled Ethernet, hardware serial, software serial and via USB to serial converters.

I hope you enjoyed this demo. Visit us on <u>www.sylvestersolutions.com</u> for more demos.