Index	Class	Type	Name	Description	
1	**Address**	**WSLUA_CLASS_DEFINE**	**Address**	**/* Represents an address. */**	**/* WSLUA_CONTINUE_MODULE Pinfo */**
2	Address	WSLUA_CONSTRUCTOR	Address.ether(eth)	/* Creates an Address Object representing an Ethernet address. */	
3	Address	WSLUA_CONSTRUCTOR	Address.ip(hostname)	/* Creates an Address Object representing an IPv4 address. */	
4	Address	WSLUA_CONSTRUCTOR	Address.ipv4(hostname)	Alias - /* Creates an Address Object representing an IPv4 address. */	
5	Address	WSLUA_CONSTRUCTOR	Address.ipv6(hostname)	/* Creates an Address Object representing an IPv6 address. */	
6	Address	WSLUA_METAMETHOD	address:__tostring()	/* The string representing the address. */	
7	Address	WSLUA_METAMETHOD	address:__eq()	/* Compares two Addresses. */	
8	Address	WSLUA_METAMETHOD	Address__gc		
9	Address	WSLUA_METAMETHOD	address:__le()	/* Compares two Addresses. */	
10	Address	WSLUA_METAMETHOD	address:__lt()	/* Compares two Addresses. */	
11					
12	**ByteArray**	**WSLUA_CLASS_DEFINE**	**ByteArray**		**/* WSLUA_CONTINUE_MODULE Tvb */**
13	ByteArray	WSLUA_CONSTRUCTOR	ByteArray.new([hexbytes], [separator])	Creates a new ByteArray object.	
14	ByteArray	WSLUA_METAMETHOD	bytearray:__tostring()	A hex-ascii string representation of the ByteArray.	
15	ByteArray	WSLUA_METAMETHOD	__call	ByteArray_subset	
16	ByteArray	WSLUA_METAMETHOD	bytearray:__concat(first, second)	The new composite ByteArray.	
17	ByteArray	WSLUA_METAMETHOD	bytearray:__eq(first, second)	Compares two ByteArray values.	
18	ByteArray	WSLUA_METAMETHOD	ByteArray__gc		
19	ByteArray	WSLUA_METHOD	bytearray:append(appended)	Append a ByteArray to this ByteArray.	
20	ByteArray	WSLUA_METHOD	bytearray:base64_decode()	Obtain a Base64 decoded ByteArray.	
21	ByteArray	WSLUA_METHOD	bytearray:get_index(index)	Get the value of a byte in a ByteArray.	
22	ByteArray	WSLUA_METHOD	bytearray:len()	Obtain the length of a ByteArray.	
23	ByteArray	WSLUA_METHOD	bytearray:prepend(prepended)	Prepend a ByteArray to this ByteArray.	
24	ByteArray	WSLUA_METHOD	bytearray:raw([offset], [length])	A Lua string of the binary bytes in the ByteArray.	
25	ByteArray	WSLUA_METHOD	bytearray:set_index(index, value)	Sets the value of an index of a ByteArray.	
26	ByteArray	WSLUA_METHOD	bytearray:set_size(size)	Sets the size of a ByteArray, either truncating it or filling it with zeros.	
27	ByteArray	WSLUA_METHOD	bytearray:subset(offset, length)	A ByteArray containing the requested segment.	
28	ByteArray	WSLUA_METHOD	bytearray:tohex([lowercase], [separator])	A hex-ascii string representation of the ByteArray.	
29	ByteArray	WSLUA_METHOD	bytearray:tvb(name)	The created Tvb.	
30					
31	**CaptureInfo**	**WSLUA_CLASS_DEFINE**	**CaptureInfo**	**passed into Lua as an argument by `FileHandler` callback "read" functions**	**/* WSLUA_CONTINUE_MODULE File */**
32	CaptureInfo	WSLUA_METAMETHOD	captureinfo:__tostring()	String of debug information.	
33	CaptureInfo	WSLUA_METAMETHOD	CaptureInfo__gc		
34	CaptureInfo	WSLUA_ATTRIBUTE_RWREG	captureinfo.comment	A string comment for the whole capture file, or nil if there is no comment.	
35	CaptureInfo	WSLUA_ATTRIBUTE_RWREG	captureinfo.encap	The packet encapsulation type for the whole file.	
36	CaptureInfo	WSLUA_ATTRIBUTE_RWREG	captureinfo.hardware	description of the hardware used to create the capture	
37	CaptureInfo	WSLUA_ATTRIBUTE_WOREG	captureinfo.hosts	Sets resolved ip-to-hostname information.	
38	CaptureInfo	WSLUA_ATTRIBUTE_RWREG	captureinfo.os	the name of the operating system used to create the capture,	
39	CaptureInfo	WSLUA_ATTRIBUTE_RWREG	captureinfo.private_table	A private Lua value unique to this file.	
40	CaptureInfo	WSLUA_ATTRIBUTE_RWREG	captureinfo.snapshot_length	The maximum packet length that could be recorded.	
41	CaptureInfo	WSLUA_ATTRIBUTE_RWREG	captureinfo.time_precision	The precision of the packet timestamps in the file.	
42	CaptureInfo	WSLUA_ATTRIBUTE_RWREG	captureinfo.user_app	the name of the application used to create the capture	
43					
44	**CaptureInfoConst**	**WSLUA_CLASS_DEFINE**	**CaptureInfoConst**	**passed into Lua as an argument by `FileHandler` callback "write" function**	**/* WSLUA_CONTINUE_MODULE File */**
45	CaptureInfoConst	WSLUA_METAMETHOD	captureinfoconst:__tostring()	String of debug information.	
46	CaptureInfoConst	WSLUA_METAMETHOD	CaptureInfoConst__gc		
47	CaptureInfoConst	WSLUA_ATTRIBUTE_ROREG	captureinfoconst.comment	A comment for the whole capture file,	
48	CaptureInfoConst	WSLUA_ATTRIBUTE_ROREG	captureinfoconst.encap	The packet encapsulation type for the whole file.	
49	CaptureInfoConst	WSLUA_ATTRIBUTE_ROREG	captureinfoconst.hardware	the description of the hardware used to create the capture,	
50	CaptureInfoConst	WSLUA_ATTRIBUTE_ROREG	captureinfoconst.hosts	A ip-to-hostname Lua table of two key-ed names:	
51	CaptureInfoConst	WSLUA_ATTRIBUTE_ROREG	captureinfoconst.os	the name of the operating system used to create the capture,	
52	CaptureInfoConst	WSLUA_ATTRIBUTE_RWREG	captureinfoconst.private_table	A private Lua value unique to this file.	
53	CaptureInfoConst	WSLUA_ATTRIBUTE_ROREG	captureinfoconst.snapshot_length	The maximum packet length that is actually recorded	
54	CaptureInfoConst	WSLUA_ATTRIBUTE_ROREG	captureinfoconst.type	The file type.	
55	CaptureInfoConst	WSLUA_ATTRIBUTE_ROREG	captureinfoconst.user_app	the name of the application used to create the capture	
56					
57	**Column**	**WSLUA_CLASS_DEFINE**	**Column**	**A Column in the packet list.**	**/* WSLUA_CONTINUE_MODULE Pinfo */**
58	Column	WSLUA_METAMETHOD	column:__tostring()	The column's string text (in parenthesis if not available).	
59	Column	WSLUA_METAMETHOD	Column__gc		
60	Column	WSLUA_METHOD	column:append(text)	Appends text to a Column.	
61	Column	WSLUA_METHOD	column:clear()	Clears a Column.	
62	Column	WSLUA_METHOD	column:clear_fence()	Clear Column text fence.	
63	Column	WSLUA_METHOD	column:fence()	Sets Column text fence, to prevent overwriting.	

Index	Class	Type	Name	Description	
64	Column	WSLUA_METHOD	column:prepend(text)	Prepends text to a Column.	
65	Column	WSLUA_METHOD	column:preppend(text)	Alias - Prepends text to a Column.	
66	Column	WSLUA_METHOD	column:set(text)	Sets the text of a Column.	
67					
68	**Columns**	**WSLUA_CLASS_DEFINE**	**Columns**	**The Columns of the packet list.**	**/* WSLUA_CONTINUE_MODULE Pinfo */**
69	Columns	WSLUA_METAMETHOD	columns:__tostring()	The string "Columns". This has no real use aside from debugging.	
70	Columns	WSLUA_METAMETHOD	Columns__gc		
71	Columns	WSLUA_METAMETHOD	columns:__index()	Get a specific Column.	
72	Columns	WSLUA_METAMETHOD	columns:__newindex(column, text)	Sets the text of a specific column.	
73					
74	**Dir**	**WSLUA_CLASS_DEFINE**	**Dir**	**A Directory object, as well as associated functions.**	**/* WSLUA_MODULE Dir Directory Handling Functions ***
75	Dir	WSLUA_CONSTRUCTOR	Dir.exists(name)	Boolean true if the directory exists, false if it's a file, nil on error or not-exist.	
76	Dir	WSLUA_CONSTRUCTOR	Dir.global_config_path([filename])	Gets the global configuration directory path, with filename if supplied.	
77	Dir	WSLUA_CONSTRUCTOR	Dir.global_plugins_path()	Gets the global plugins directory path.	
78	Dir	WSLUA_CONSTRUCTOR	Dir.make(name)	Creates a directory.	
79	Dir	WSLUA_CONSTRUCTOR	Dir.open(pathname, [extension])	Opens a directory and returns a Dir object representing the files in the directory.	
80	Dir	WSLUA_CONSTRUCTOR	Dir.personal_config_path([filename])	Gets the personal configuration directory path, with filename if supplied.	
81	Dir	WSLUA_CONSTRUCTOR	Dir.personal_plugins_path()	Gets the personal plugins directory path.	
82	Dir	WSLUA_CONSTRUCTOR	Dir.remove(name)	Removes an empty directory.	
83	Dir	WSLUA_CONSTRUCTOR	Dir.remove_all(name)	Removes an empty or non-empty directory.	
84	Dir	WSLUA_METAMETHOD	Dir__gc		
85	Dir	WSLUA_METAMETHOD	dir:__call()	Gets the next file or subdirectory within the directory, or nil when done.	
86	Dir	WSLUA_METHOD	dir:close()	Closes the directory. Called automatically during garbage collection of a Dir object.	
87					
88	**Dissector**	**WSLUA_CLASS_DEFINE**	**Dissector**	**A refererence to a dissector, used to call a dissector against a packet or a part of it.**	**/* WSLUA_CONTINUE_MODULE Proto */**
89	Dissector	WSLUA_CONSTRUCTOR	Dissector.get(name)	The Dissector reference if found, otherwise nil.	
90	Dissector	WSLUA_CONSTRUCTOR	Dissector.list()	Gets a Lua array table of all registered Dissector names.	
91	Dissector	WSLUA_METAMETHOD	dissector:__tostring()	A string of the protocol's short name.	
92	Dissector	WSLUA_METAMETHOD	Dissector__gc		
93	Dissector	WSLUA_METAMETHOD	dissector:__call(tvb, pinfo, tree)	Return description missing from wsluarm. Add ???	
94	Dissector	WSLUA_METHOD	dissector:call(tvb, pinfo, tree)	Calls a dissector against a given packet (or part of it).	
95					
96	**DissectorTable**	**WSLUA_CLASS_DEFINE**	**DissectorTable**	**A table of subdissectors of a particular protocol**	**/* WSLUA_CONTINUE_MODULE Proto */**
97	DissectorTable	WSLUA_CONSTRUCTOR	DissectorTable.get(tablename)	Obtain a reference to an existing dissector table.	
98	DissectorTable	WSLUA_CONSTRUCTOR	DissectorTable.heuristic_list()	Gets a Lua array table of all heuristic list names	
99	DissectorTable	WSLUA_CONSTRUCTOR	DissectorTable.list()	Gets a Lua array table of all DissectorTable names	
100	DissectorTable	WSLUA_CONSTRUCTOR	DissectorTable.new(tablename, [uiname], [type], [base], [proto])	Creates a new DissectorTable for your dissector's use.	
101	DissectorTable	WSLUA_CONSTRUCTOR	DissectorTable.try_heuristics(listname, tvb, pinfo, tree)	Try all the dissectors in a given heuristic dissector table.	
102	DissectorTable	WSLUA_METAMETHOD	dissectortable:__tostring()	A string of debug information about the DissectorTable.	
103	DissectorTable	WSLUA_METAMETHOD	DissectorTable__gc		
104	DissectorTable	WSLUA_METHOD	dissectortable:add(pattern, dissector)	Add a Proto with a dissector function or a Dissector object to the dissector table.	
105	DissectorTable	WSLUA_METHOD	dissectortable:add_for_decode_as(proto)	Add the given Proto to the "Decode as…" list for this DissectorTable	
106	DissectorTable	WSLUA_METHOD	dissectortable:get_dissector(pattern)	Try to obtain a dissector from a table.	
107	DissectorTable	WSLUA_METHOD	dissectortable:remove(pattern, dissector)	Remove a dissector or a range of dissectors from a table.	
108	DissectorTable	WSLUA_METHOD	dissectortable:remove_all(dissector)	Remove all dissectors from a table.	
109	DissectorTable	WSLUA_METHOD	dissectortable:set(pattern, dissector)	Clear all existing dissectors from a table and add a new dissector or a range of new dissectors.	
110	DissectorTable	WSLUA_METHOD	dissectortable:try(pattern, tvb, pinfo, tree)	Try to call a dissector from a table.	
111					
112	**PseudoHeader**	**WSLUA_CLASS_DEFINE**	**PseudoHeader**	**A pseudoheader to be used to save captured frames.**	**/* WSLUA_MODULE Dumper Saving Capture Files**
113	PseudoHeader	WSLUA_CONSTRUCTOR	PseudoHeader.none()	Creates a "no" pseudoheader.	
114	PseudoHeader	WSLUA_CONSTRUCTOR	PseudoHeader.eth([fcslen])	Creates an ethernet pseudoheader.	
115	PseudoHeader	WSLUA_CONSTRUCTOR	PseudoHeader.atm([aal], [vpi], [vci], [channel], [cells], [aal5u2u], [aal5len])	Creates an ATM pseudoheader.	
116	PseudoHeader	WSLUA_CONSTRUCTOR	PseudoHeader.mtp2([sent], [annexa], [linknum])	The MTP2 pseudoheader	
117	PseudoHeader	WSLUA_METAMETHOD	PseudoHeader__gc		
118					
119	**Dumper**	**WSLUA_CLASS_DEFINE**	**Dumper**		**/* WSLUA_MODULE Dumper Saving Capture Files**
120	Dumper	WSLUA_CONSTRUCTOR	Dumper.new(filename, [filetype], [encap])	Creates a file to write packets. Dumper:new_for_current() will probably be a better choice.	
121	Dumper	WSLUA_METHOD	dumper:close()	Closes a dumper.	
122	Dumper	WSLUA_METHOD	dumper:flush()	Writes all unsaved data of a dumper to the disk.	
123	Dumper	WSLUA_METHOD	dumper:dump(timestamp, pseudoheader, bytearray)	Dumps an arbitrary packet. Note: Dumper:dump_current() will fit best in most cases.	
124	Dumper	WSLUA_METHOD	dumper:new_for_current([filetype])	Creates a capture file using the same encapsulation as the one of the current packet.	
125	Dumper	WSLUA_METHOD	dumper:dump_current()	Dumps the current packet as it is.	
126	Dumper	WSLUA_METAMETHOD	Dumper__gc		

Index	Class	Type	Name	Description	
127					
128	**FieldInfo**	**WSLUA_CLASS_DEFINE**	**FieldInfo**	**An extracted Field from dissected packet data.**	**/* WSLUA_MODULE Field Obtaining Dissection Data */**
129	FieldInfo	WSLUA_ATTRIBUTE	fieldinfo.len	RO The length of this field.	
130	FieldInfo	WSLUA_METAMETHOD	fieldinfo:__len()	Obtain the Length of the field	
131	FieldInfo	WSLUA_ATTRIBUTE	fieldinfo.offset	RO The offset of this field.	
132	FieldInfo	WSLUA_METAMETHOD	fieldinfo:__unm()	Obtain the Offset of the field	
133	FieldInfo	WSLUA_ATTRIBUTE	fieldinfo.value	RO The value of this field.	
134	FieldInfo	WSLUA_METAMETHOD	fieldinfo:__call()	Obtain the Value of the field.	
135	FieldInfo	WSLUA_ATTRIBUTE	fieldinfo.label	RO The string representing this field.	
136	FieldInfo	WSLUA_METAMETHOD	fieldinfo:__tostring()	The string representation of the field.	
137	FieldInfo	WSLUA_ATTRIBUTE	fieldinfo.display	RO The string display of this field as seen in GUI.	
138	FieldInfo	WSLUA_ATTRIBUTE	fieldinfo.type	RO The internal field type, a number which matches one of the ftype values in init.lua.	
139	FieldInfo	WSLUA_ATTRIBUTE	fieldinfo.source	RO The source Tvb object the FieldInfo is derived from, or nil if there is none.	
140	FieldInfo	WSLUA_ATTRIBUTE	fieldinfo.range	RO The TvbRange covering the bytes of this field in a Tvb or nil if there is none.	
141	FieldInfo	WSLUA_ATTRIBUTE	fieldinfo.tvb	RO The TvbRange covering the bytes of this field in a Tvb or nil if there is none.	ALIAS ???
142	FieldInfo	WSLUA_ATTRIBUTE	fieldinfo.generated	RO Whether this field was marked as generated (boolean).	
143	FieldInfo	WSLUA_ATTRIBUTE	fieldinfo.hidden	RO Whether this field was marked as hidden (boolean).	
144	FieldInfo	WSLUA_ATTRIBUTE	fieldinfo.is_url	RO Whether this field was marked as being a URL (boolean).	
145	FieldInfo	WSLUA_ATTRIBUTE	fieldinfo.little_endian	RO Whether this field is little-endian encoded (boolean).	
146	FieldInfo	WSLUA_ATTRIBUTE	fieldinfo.big_endian	RO Whether this field is big-endian encoded (boolean).	
147	FieldInfo	WSLUA_ATTRIBUTE	fieldinfo.name	RO The filter name of this field.	
148	FieldInfo	WSLUA_METAMETHOD	fieldinfo:__eq()	Checks whether lhs is within rhs.	
149	FieldInfo	WSLUA_METAMETHOD	fieldinfo:__le()	Checks whether the end byte of lhs is before the end of rhs.	
150	FieldInfo	WSLUA_METAMETHOD	fieldinfo:__lt()	Checks whether the end byte of rhs is before the beginning of rhs.	
151	FieldInfo	WSLUA_METAMETHOD	FieldInfo__gc		
152					
153	Global (Field)	WSLUA_FUNCTION	all_field_infos()	Obtain all fields from the current tree.	/* WSLUA_MODULE Field Obtaining Dissection Data */
154					
155	**Field**	**WSLUA_CLASS_DEFINE**	**Field**	**A Field extractor to obtain field values.**	**/* WSLUA_MODULE Field Obtaining Dissection Data */**
156	Field	WSLUA_CONSTRUCTOR	Field.new(fieldname)	Create a Field extractor.	
157	Field	WSLUA_CONSTRUCTOR	Field.list()	Gets a Lua array table of all registered field filter names.	
158	Field	WSLUA_ATTRIBUTE	field.name	RO The filter name of this field, or nil.	
159	Field	WSLUA_ATTRIBUTE	field.display	RO The full display name of this field, or nil.	
160	Field	WSLUA_ATTRIBUTE	field.type	RO The `ftype` of this field, or nil.	
161	Field	WSLUA_METAMETHOD	field:__call()	Obtain all values (see FieldInfo) for this field.	
162	Field	WSLUA_METAMETHOD	field:__tostring()	Obtain a string with the field filter name.	
163	Field	WSLUA_METAMETHOD	Field__gc		
164					
165	**FileHandler**	**WSLUA_CLASS_DEFINE**	**FileHandler**	**A FileHandler object, created by a call to FileHandler.new(arg1, arg2, …).**	**/* WSLUA_CONTINUE_MODULE File */**
166	FileHandler	WSLUA_CONSTRUCTOR	FileHandler.new(description, name, internal_description, type)	Creates a new FileHandler	
167	FileHandler	WSLUA_METAMETHOD	filehandler:__tostring()	Generates a string of debug info for the FileHandler	
168	FileHandler	WSLUA_METAMETHOD	FileHandler__gc		
169	FileHandler	WSLUA_ATTRIBUTE	filehandler.read_open	WO The Lua function to be called when Wireshark opens a file for reading.	
170	FileHandler	WSLUA_ATTRIBUTE	filehandler.read	WO The Lua function to be called when Wireshark wants to read a packet from the file.	
171	FileHandler	WSLUA_ATTRIBUTE	filehandler.seek_read	WO The Lua function to be called when Wireshark wants to read a packet from the file at the given offset.	
172	FileHandler	WSLUA_ATTRIBUTE	filehandler.read_close	WO The Lua function to be called when Wireshark wants to close the read file completely.	
173	FileHandler	WSLUA_ATTRIBUTE	filehandler.seq_read_close	WO The Lua function to be called when Wireshark wants to close the sequentially-read file.	
174	FileHandler	WSLUA_ATTRIBUTE	filehandler.can_write_encap	WO The Lua function to be called when Wireshark wants to write a file, by checking if this file writer can handle the wtap packet encapsulation(s).	
175	FileHandler	WSLUA_ATTRIBUTE	filehandler.write_open	WO The Lua function to be called when Wireshark opens a file for writing.	
176	FileHandler	WSLUA_ATTRIBUTE	filehandler.write	WO The Lua function to be called when Wireshark wants to write a packet to the file.	
177	FileHandler	WSLUA_ATTRIBUTE	filehandler.write_finish	WO The Lua function to be called when Wireshark wants to close the written file.	**Typo ???**
178	FileHandler	WSLUA_ATTRIBUTE	filehandler.type	RO The internal file type.	
179	FileHandler	WSLUA_ATTRIBUTE	filehandler.extensions	RW One or more semicolon-separated file extensions that this file type usually uses.	
180	FileHandler	WSLUA_ATTRIBUTE	filehandler.writing_must_seek	RW true if the ability to seek is required when writing this file format, else false.	
181	FileHandler	WSLUA_ATTRIBUTE	filehandler.writes_name_resolution	RW true if the file format supports name resolution records, else false.	
182	FileHandler	WSLUA_ATTRIBUTE	filehandler.supported_comment_types	RW set to the bit-wise OR'ed number representing the type of comments the file writer supports writing, based on the numbers in the `wtap_comments` table.	
183					
184	Global (FileHandler)	WSLUA_FUNCTION	register_filehandler(filehandler)	Register the FileHandler into Wireshark/TShark, so they can read/write this new format.	/* WSLUA_CONTINUE_MODULE File */
185	Global (FileHandler)	WSLUA_FUNCTION	deregister_filehandler(filehandler)	Deregister the FileHandler from Wireshark/TShark, so it no longer gets used for reading/writing/display.	/* WSLUA_CONTINUE_MODULE File */
186					
187	**File**	**WSLUA_CLASS_DEFINE**	**File**	**A File object, passed into Lua as an argument by FileHandler callback functions (e.g., read_open, read, write, etc.).**	**/* WSLUA_MODULE File Custom File Format Reading A**

Index	Class	Type	Name	Description	
188	File	WSLUA_METHOD	file:read()	Reads from the File, similar to Lua's file:read(). See Lua 5.x ref manual for file:read().	
189	File	WSLUA_METHOD	file:seek()	Seeks in the File, similar to Lua's file:seek(). See Lua 5.x ref manual for file:seek().	
190	File	WSLUA_METHOD	file:lines()	Lua iterator function for retrieving ASCII File lines, similar to Lua's file:lines().	
191	File	WSLUA_METHOD	file:write()	Writes to the File, similar to Lua's file:write(). See Lua 5.x ref manual for file:write().	
192	File	WSLUA_METAMETHOD	file:__tostring()	Generates a string of debug info for the File object	
193	File	WSLUA_METAMETHOD	File__gc		
194	File	WSLUA_ATTRIBUTE	file.compressed	RO Whether the File is compressed or not.	
195					
196	**FrameInfo**	**WSLUA_CLASS_DEFINE**	**FrameInfo**	**This object represents frame data and meta-data (data about the frame/packet) for a given read/seek_read/write's frame.**	**/* WSLUA_CONTINUE_MODULE File */**
197	FrameInfo	WSLUA_METAMETHOD	frameinfo:__tostring()	Generates a string of debug info for the FrameInfo	
198	FrameInfo	WSLUA_METAMETHOD	FrameInfo__gc		
199	FrameInfo	WSLUA_METHOD	frameinfo:read_data(file, length)	Tells Wireshark to read directly from given file into frame data buffer, for length bytes.	
200	FrameInfo	WSLUA_ATTRIBUTE	frameinfo.comment	RW table of comments in this frame.	
201	FrameInfo	WSLUA_ATTRIBUTE	frameinfo.time	RW The packet timestamp as an NSTime object.	
202	FrameInfo	WSLUA_ATTRIBUTE	frameinfo.data	RW The data buffer containing the packet.	
203	FrameInfo	WSLUA_ATTRIBUTE	frameinfo.rec_type	RW The record type of the packet frame	
204	FrameInfo	WSLUA_ATTRIBUTE	frameinfo.flags	RW The presence flags of the packet frame.	
205	FrameInfo	WSLUA_ATTRIBUTE	frameinfo.captured_length	RW The captured packet length, and thus the length of the buffer passed to the `FrameInfo.data` field.	
206	FrameInfo	WSLUA_ATTRIBUTE	frameinfo.original_length	RW The on-the-wire packet length, which may be longer than the `captured_length`.	
207	FrameInfo	WSLUA_ATTRIBUTE	frameinfo.encap	RW The packet encapsulation type for the frame/packet, if the file supports per-packet types.	
208					
209	**FrameInfoConst**	**WSLUA_CLASS_DEFINE**	**FrameInfoConst**	**This has similar attributes/properties as FrameInfo, but the fields can only be read from, not written to.**	**/* WSLUA_CONTINUE_MODULE File */**
210	FrameInfoConst	WSLUA_METAMETHOD	frameinfoconst:__tostring()	Generates a string of debug info for the FrameInfo	
211	FrameInfoConst	WSLUA_METAMETHOD	FrameInfoConst__gc		
212	FrameInfoConst	WSLUA_METHOD	frameinfoconst:write_data(file, [length])	Tells Wireshark to write directly to given file from the frame data buffer, for length bytes.	
213	FrameInfoConst	WSLUA_ATTRIBUTE	frameinfoconst.comment	RO The first string comment for the packet, if any; nil if there is no comment.	
214	FrameInfoConst	WSLUA_ATTRIBUTE	frameinfoconst.time	RO The packet timestamp as an NSTime object.	
215	FrameInfoConst	WSLUA_ATTRIBUTE	frameinfoconst.data	RO The data buffer containing the packet.	
216	FrameInfoConst	WSLUA_ATTRIBUTE	frameinfoconst.rec_type	RO The record type of the packet frame	**Typo ???**
217	FrameInfoConst	WSLUA_ATTRIBUTE	frameinfoconst.flags	RO The presence flags of the packet frame - see `wtap_presence_flags` in `init.lua` for bits.	
218	FrameInfoConst	WSLUA_ATTRIBUTE	frameinfoconst.captured_length	RO The captured packet length,	
219	FrameInfoConst	WSLUA_ATTRIBUTE	frameinfoconst.original_length	RO The on-the-wire packet length,	
220	FrameInfoConst	WSLUA_ATTRIBUTE	frameinfoconst.encap	RO The packet encapsulation type, if the file supports per-packet types.	
221					
222	Global (Gui)	WSLUA_FUNCTION	gui_enabled()	Checks if we're running inside a GUI (i.e. Wireshark) or not.	/* WSLUA_MODULE Gui GUI Support */
223	Global (Gui)	WSLUA_FUNCTION	register_menu(name, action, [group])	Register a menu item in one of the main menus. Requires a GUI.	
224	Global (Gui)	WSLUA_FUNCTION	new_dialog(title, action, …)	Displays a dialog, prompting for input.	Typo - "labeld"
225					
226	**ProgDlg**	**WSLUA_CLASS_DEFINE**	**ProgDlg**	**Creates and manages a modal progress bar.**	**/* WSLUA_MODULE Gui GUI Support */**
227	ProgDlg	WSLUA_CONSTRUCTOR	ProgDlg.new([title], [task])	Creates and displays a new `ProgDlg` progress bar with a btn:[Cancel] button and optional title.	
228	ProgDlg	WSLUA_METHOD	progdlg:update(progress, [task])	Sets the progress dialog's progress bar position based on percentage done.	
229	ProgDlg	WSLUA_METHOD	progdlg:stopped()	Checks whether the user has pressed the btn:[Cancel] button.	
230	ProgDlg	WSLUA_METHOD	progdlg:close()	Hides the progress bar.	
231	ProgDlg	WSLUA_METAMETHOD	ProgDlg__tostring	A string specifying whether the Progress Dialog has stopped or not.	Document ???
232	ProgDlg	WSLUA_METAMETHOD	ProgDlg__gc		
233					
234	**TextWindow**	**WSLUA_CLASS_DEFINE**	**TextWindow**	**Creates and manages a text window.**	**/* WSLUA_MODULE Gui GUI Support */**
235	TextWindow	WSLUA_CONSTRUCTOR	TextWindow.new([title])	Creates a new TextWindow text window and displays it. Requires a GUI.	
236	TextWindow	WSLUA_METHOD	textwindow:set_atclose(action)	Set the function that will be called when the text window closes.	
237	TextWindow	WSLUA_METHOD	textwindow:set(text)	Sets the text to be displayed.	
238	TextWindow	WSLUA_METHOD	textwindow:append(text)	Appends text to the current window contents.	
239	TextWindow	WSLUA_METHOD	textwindow:prepend(text)	Prepends text to the current window contents.	
240	TextWindow	WSLUA_METHOD	textwindow:clear()	Erases all of the text in the window.	
241	TextWindow	WSLUA_METHOD	textwindow:get_text()	Get the text of the window.	
242	TextWindow	WSLUA_METHOD	textwindow:close()	Close the window.	
243	TextWindow	WSLUA_METHOD	textwindow:set_editable([editable])	Make this text window editable.	
244	TextWindow	WSLUA_METHOD	textwindow:add_button(label, function)	Adds a button with an action handler to the text window.	
245	TextWindow	WSLUA_METAMETHOD	TextWindow__gc		
246	TextWindow	WSLUA_METAMETHOD	{"__tostring", TextWindow_get_text},	Get the text of the window.	
247					
248	Global (Gui)	WSLUA_FUNCTION	retap_packets()	Rescans all packets and runs each tap listener without reconstructing the display.	/* WSLUA_MODULE Gui GUI Support */

Index	Class	Type	Name	Description	
249	Global (Gui)	WSLUA_FUNCTION	copy_to_clipboard(text)	Copy a string into the clipboard. Requires a GUI.	
250	Global (Gui)	WSLUA_FUNCTION	open_capture_file(filename, filter)	Open and display a capture file. Requires a GUI.	
251	Global (Gui)	WSLUA_FUNCTION	get_filter()	Get the main filter text.	
252	Global (Gui)	WSLUA_FUNCTION	set_filter(text)	Set the main filter text.	
253	Global (Gui)	WSLUA_FUNCTION	get_color_filter_slot(row)	Gets the current packet coloring rule (by index) for the current session.	
254	Global (Gui)	WSLUA_FUNCTION	set_color_filter_slot(row, text)	Sets a packet coloring rule (by index) for the current session.	
255	Global (Gui)	WSLUA_FUNCTION	apply_filter()	Apply the filter in the main filter box. Requires a GUI.	
256	Global (Gui)	WSLUA_FUNCTION	reload()	Reload the current capture file. Deprecated. Use reload_packets() instead.	
257	Global (Gui)	WSLUA_FUNCTION	reload_packets()	Reload the current capture file. Requires a GUI.	
258	Global (Gui)	WSLUA_FUNCTION	redissect_packets()	Redissect all packets in the current capture file. Requires a GUI.	
259	Global (Gui)	WSLUA_FUNCTION	reload_lua_plugins()	Reload all Lua plugins.	
260	Global (Gui)	WSLUA_FUNCTION	browser_open_url(url)	Opens an URL in a web browser. Requires a GUI.	
261	Global (Gui)	WSLUA_FUNCTION	browser_open_data_file(filename)	Open a file located in the data directory (specified in the Wireshark preferences) in the web browser.	
262					
263	**Listener**	**WSLUA_CLASS_DEFINE**	**Listener**	**A `Listener` is called once for every packet that matches a certain filter or has a certain tap.**	**/* WSLUA_MODULE Listener Post-Dissection Packet A**
264	Listener	WSLUA_CONSTRUCTOR	Listener.new([tap], [filter], [allfields])	Creates a new Listener tap object.	
265	Listener	WSLUA_CONSTRUCTOR	Listener.list()	Gets a Lua array table of all registered Listener tap names.	
266	Listener	WSLUA_METHOD	listener:remove()	Removes a tap Listener.	
267	Listener	WSLUA_METAMETHOD	listener:__tostring()	Generates a string of debug info for the tap Listener.	
268	Listener	WSLUA_METAMETHOD	Listener__gc		
269	Listener	WSLUA_ATTRIBUTE	listener.packet	WO A function that will be called once every packet matches the `Listener` listener filter.	
270	Listener	WSLUA_ATTRIBUTE	listener.draw	WO A function that will be called once every few seconds to redraw the GUI objects; in TShark this funtion is called only at the very end of the capture file.	
271	Listener	WSLUA_ATTRIBUTE	listener.reset	WO A function that will be called at the end of the capture run.	
272					
273	**NSTime**	**WSLUA_CLASS_DEFINE**	**NSTime**	**NSTime represents a nstime_t. This is an object with seconds and nanoseconds.**	**/* WSLUA_CONTINUE_MODULE Pinfo */**
274	NSTime	WSLUA_CONSTRUCTOR	NSTime.new([seconds], [nseconds])	Creates a new NSTime object.	
275	NSTime	WSLUA_METAMETHOD	nstime:__call([seconds], [nseconds])	Creates a NSTime object.	
276	NSTime	WSLUA_METHOD	nstime:tonumber()	Returns a Lua number of the NSTime representing seconds from epoch	
277	NSTime	WSLUA_METAMETHOD	nstime:__tostring()	The string representing the nstime.	
278	NSTime	WSLUA_METAMETHOD	nstime:__add()	Calculates the sum of two NSTimes.	
279	NSTime	WSLUA_METAMETHOD	nstime:__sub()	Calculates the diff of two NSTimes.	
280	NSTime	WSLUA_METAMETHOD	nstime:__unm()	Calculates the negative NSTime.	
281	NSTime	WSLUA_METAMETHOD	nstime:__eq()	Compares two NSTimes.	
282	NSTime	WSLUA_METAMETHOD	nstime:__le()	Compares two NSTimes.	
283	NSTime	WSLUA_METAMETHOD	nstime:__lt()	Compares two NSTimes.	
284	NSTime	WSLUA_METAMETHOD	NSTime__gc		
285	NSTime	WSLUA_ATTRIBUTE	nstime.secs	RW The NSTime seconds.	
286	NSTime	WSLUA_ATTRIBUTE	nstime.nsecs	RW The NSTime nano seconds.	
287					
288	**PrivateTable**	**WSLUA_CLASS_DEFINE**	**PrivateTable**	**PrivateTable represents the pinfo->private_table.**	**/* WSLUA_MODULE Pinfo Obtaining Packet Informatio**
289	PrivateTable	WSLUA_METAMETHOD	privatetable:__tostring()	Gets debugging type information about the private table.	
290	PrivateTable	WSLUA_METAMETHOD	PrivateTable__index	Gets the text of a specific entry.	
291	PrivateTable	WSLUA_METAMETHOD	PrivateTable__newindex	Sets the text of a specific entry.	
292	PrivateTable	WSLUA_METAMETHOD	PrivateTable__gc		
293					
294	**Pinfo**	**WSLUA_CLASS_DEFINE**	**Pinfo**	**Packet information.**	**/* WSLUA_MODULE Pinfo Obtaining Packet Informatio**
295	Pinfo	WSLUA_METAMETHOD	Pinfo__tostring		
296	Pinfo	WSLUA_ATTRIBUTE	pinfo.visited	RO Whether this packet has been already visited.	
297	Pinfo	WSLUA_ATTRIBUTE	pinfo.number	RO The number of this packet in the current file.	
298	Pinfo	WSLUA_ATTRIBUTE	pinfo.len	RO The length of the frame.	
299	Pinfo	WSLUA_ATTRIBUTE	pinfo.caplen	RO The captured length of the frame.	
300	Pinfo	WSLUA_ATTRIBUTE	pinfo.abs_ts	RO When the packet was captured.	
301	Pinfo	WSLUA_ATTRIBUTE	pinfo.rel_ts	RO Number of seconds passed since beginning of capture.	
302	Pinfo	WSLUA_ATTRIBUTE	pinfo.delta_ts	RO Number of seconds passed since the last captured packet.	
303	Pinfo	WSLUA_ATTRIBUTE	pinfo.delta_dis_ts	RO Number of seconds passed since the last displayed packet.	
304	Pinfo	WSLUA_ATTRIBUTE	pinfo.curr_proto	RO Which Protocol are we dissecting.	
305	Pinfo	WSLUA_ATTRIBUTE	pinfo.can_desegment	RW Set if this segment could be desegmented.	
306	Pinfo	WSLUA_ATTRIBUTE	pinfo.desegment_len	RW Estimated number of additional bytes required for completing the PDU.	
307	Pinfo	WSLUA_ATTRIBUTE	pinfo.desegment_offset	RW Offset in the tvbuff at which the dissector will continue processing when next called.	
308	Pinfo	WSLUA_ATTRIBUTE	pinfo.fragmented	RO If the protocol is only a fragment.	
309	Pinfo	WSLUA_ATTRIBUTE	pinfo.in_error_pkt	RO If we're inside an error packet.	
310	Pinfo	WSLUA_ATTRIBUTE	pinfo.match_uint	RO Matched uint for calling subdissector from table.	

Index	Class	Type	Name	Description	
311	Pinfo	WSLUA_ATTRIBUTE	pinfo.match_string	RO Matched string for calling subdissector from table.	
312	Pinfo	WSLUA_ATTRIBUTE	pinfo.port_type	RW Type of Port of .src_port and .dst_port.	
313	Pinfo	WSLUA_ATTRIBUTE	pinfo.src_port	RW Source Port of this Packet.	
314	Pinfo	WSLUA_ATTRIBUTE	pinfo.dst_port	RW Destination Port of this Packet.	
315	Pinfo	WSLUA_ATTRIBUTE	pinfo.dl_src	RW Data Link Source Address of this Packet.	
316	Pinfo	WSLUA_ATTRIBUTE	pinfo.dl_dst	RW Data Link Destination Address of this Packet.	
317	Pinfo	WSLUA_ATTRIBUTE	pinfo.net_src	RW Network Layer Source Address of this Packet.	
318	Pinfo	WSLUA_ATTRIBUTE	pinfo.net_dst	RW Network Layer Destination Address of this Packet.	
319	Pinfo	WSLUA_ATTRIBUTE	pinfo.src	RW Source Address of this Packet.	
320	Pinfo	WSLUA_ATTRIBUTE	pinfo.dst	RW Destination Address of this Packet.	
321	Pinfo	WSLUA_ATTRIBUTE	pinfo.p2p_dir	RW direction of this Packet. (incoming / outgoing)	
322	Pinfo	WSLUA_ATTRIBUTE	pinfo.match	RO Port/Data we are matching.	
323	Pinfo	WSLUA_ATTRIBUTE	pinfo.columns	RO Access to the packet list columns.	
324	Pinfo	WSLUA_ATTRIBUTE	pinfo.cols	RO Access to the packet list columns (equivalent to pinfo.columns).	
325	Pinfo	WSLUA_ATTRIBUTE	pinfo.private	RO Access to the private table entries.	
326	Pinfo	WSLUA_ATTRIBUTE	pinfo.hi	RW higher Address of this Packet.	
327	Pinfo	WSLUA_ATTRIBUTE	pinfo.lo	RO lower Address of this Packet.	
328	Pinfo	WSLUA_ATTRIBUTE	pinfo.conversation	WO sets the packet conversation to the given Proto object.	
329	Pinfo	WSLUA_METAMETHOD	Pinfo__gc		
330					
331	**Pref**	**WSLUA_CLASS_DEFINE**	**Pref**	**A preference of a Proto.**	**/* WSLUA_CONTINUE_MODULE Proto */**
332	Pref	WSLUA_CONSTRUCTOR	Pref.bool(label, default, descr)	Creates a boolean preference to be added to a Proto.prefs Lua table.	
333	Pref	WSLUA_CONSTRUCTOR	Pref.uint(label, default, descr)	Creates an (unsigned) integer preference to be added to a Proto.prefs Lua table.	
334	Pref	WSLUA_CONSTRUCTOR	Pref.string(label, default, descr)	Creates a string preference to be added to a Proto.prefs Lua table.	
335	Pref	WSLUA_CONSTRUCTOR	Pref.enum(label, default, descr, enum, radio)	Creates an enum preference to be added to a Proto.prefs Lua table.	
336	Pref	WSLUA_CONSTRUCTOR	Pref.range(label, default, descr, max)	Creates a range (numeric text entry) preference to be added to a Proto.prefs Lua table.	
337	Pref	WSLUA_CONSTRUCTOR	Pref.statictext(label, descr)	Creates a static text string to be added to a Proto.prefs Lua table.	
338	Pref	WSLUA_METAMETHOD	Pref__gc		
339					
340	**Prefs**	**WSLUA_CLASS_DEFINE**	**Prefs**	**The table of preferences of a protocol.**	**/* WSLUA_CONTINUE_MODULE Proto */**
341	Prefs	WSLUA_METAMETHOD	prefs:__newindex(name, pref)	Creates a new preference.	
342	Prefs	WSLUA_METAMETHOD	prefs:__index(name)	Get the value of a preference setting.	
343	Prefs	WSLUA_METAMETHOD	Prefs__gc		
344					
345	**ProtoExpert**	**WSLUA_CLASS_DEFINE**	**ProtoExpert**	**A Protocol expert info field, to be used when adding items to the dissection tree.**	**/* WSLUA_CONTINUE_MODULE Proto */**
346	ProtoExpert	WSLUA_CONSTRUCTOR	ProtoExpert.new(abbr, text, group, severity)	Creates a new ProtoExpert object to be used for a protocol's expert information notices.	
347	ProtoExpert	WSLUA_METAMETHOD	protoexpert:__tostring()	Returns a string with debugging information about a ProtoExpert object.	
348	ProtoExpert	WSLUA_METAMETHOD	ProtoExpert__gc		
349					
350	**ProtoField**	**WSLUA_CLASS_DEFINE**	**ProtoField**	**A Protocol field (to be used when adding items to the dissection tree).**	**/* WSLUA_CONTINUE_MODULE Proto */**
351	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.new(name, abbr, type, [valuestring], [base], [mask], [descr])	Creates a new ProtoField object to be used for a protocol field.	
352	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.char(abbr, [name], [base], [valuestring], [mask], [desc])	Creates a ProtoField of an 8-bit ASCII character.	
353	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.uint8(abbr, [name], [base], [valuestring], [mask], [desc])	Creates a ProtoField of an unsigned 8-bit integer (i.e., a byte).	
354	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.uint16(abbr, [name], [base], [valuestring], [mask], [desc])	Creates a ProtoField of an unsigned 16-bit integer.	
355	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.uint24(abbr, [name], [base], [valuestring], [mask], [desc])	Creates a ProtoField of an unsigned 24-bit integer.	
356	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.uint32(abbr, [name], [base], [valuestring], [mask], [desc])	Creates a ProtoField of an unsigned 32-bit integer.	
357	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.uint64(abbr, [name], [base], [valuestring], [mask], [desc])	Creates a ProtoField of an unsigned 64-bit integer.	
358	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.int8(abbr, [name], [base], [valuestring], [mask], [desc])	Creates a ProtoField of a signed 8-bit integer (i.e., a byte).	
359	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.int16(abbr, [name], [base], [valuestring], [mask], [desc])	Creates a ProtoField of a signed 16-bit integer.	
360	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.int24(abbr, [name], [base], [valuestring], [mask], [desc])	Creates a ProtoField of a signed 24-bit integer.	
361	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.int32(abbr, [name], [base], [valuestring], [mask], [desc])	Creates a ProtoField of a signed 32-bit integer.	
362	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.int64(abbr, [name], [base], [valuestring], [mask], [desc])	Creates a ProtoField of a signed 64-bit integer.	
363	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.framenum(abbr, [name], [base], [frametype], [mask], [desc])	Creates a ProtoField for a frame number (for hyperlinks between frames).	
364	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.bool(abbr, [name], [display], [valuestring], [mask], [desc])	Creates a ProtoField for a boolean true/false value.	
365	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.absolute_time(abbr, [name], [base], [desc])	Creates a ProtoField of a time_t structure value.	
366	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.relative_time(abbr, [name], [desc])	Creates a ProtoField of a time_t structure value.	
367	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.float(abbr, [name], [valuestring], [desc])	Creates a ProtoField of a floating point number (4 bytes).	
368	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.double(abbr, [name], [valuestring], [desc])	Creates a ProtoField of a double-precision floating point (8 bytes).	
369	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.string(abbr, [name], [display], [desc])	Creates a ProtoField of a string value.	
370	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.stringz(abbr, [name], [display], [desc])	Creates a ProtoField of a zero-terminated string value.	
371	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.bytes(abbr, [name], [display], [desc])	Creates a ProtoField for an arbitrary number of bytes.	
372	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.ubytes(abbr, [name], [display], [desc])	Creates a ProtoField for an arbitrary number of unsigned bytes.	
373	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.none(abbr, [name], [desc])	Creates a ProtoField of an unstructured type.	

Index	Class	Type	Name	Description	
374	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.ipv4(abbr, [name], [desc])	Creates a ProtoField of an IPv4 address (4 bytes).	
375	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.ipv6(abbr, [name], [desc])	Creates a ProtoField of an IPv6 address (16 bytes).	
376	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.ether(abbr, [name], [desc])	Creates a ProtoField of an Ethernet address (6 bytes).	
377	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.guid(abbr, [name], [desc])	Creates a ProtoField for a Globally Unique IDentifier (GUID).	
378	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.oid(abbr, [name], [desc])	Creates a ProtoField for an ASN.1 Organizational IDentified (OID).	
379	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.protocol(abbr, [name], [desc])	Creates a ProtoField for a sub-protocol. Since 1.99.9.	
380	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.rel_oid(abbr, [name], [desc])	Creates a ProtoField for an ASN.1 Relative-OID.	
381	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.systemid(abbr, [name], [desc])	Creates a ProtoField for an OSI System ID.	
382	ProtoField	WSLUA_CONSTRUCTOR	ProtoField.eui64(abbr, [name], [desc])	Creates a ProtoField for an EUI64.	
383	ProtoField	WSLUA_METAMETHOD	protofield:__tostring()	Returns a string with info about a protofield (for debugging purposes).	
384	ProtoField	WSLUA_METAMETHOD	ProtoField__gc		
385					
386	**Proto**	**WSLUA_CLASS_DEFINE**	**Proto**	**A new protocol in Wireshark.**	**/* WSLUA_MODULE Proto Functions For New Protocol**
387	Proto	WSLUA_CONSTRUCTOR	Proto.new(name, desc)	Creates a new Proto object.	
388	Proto	WSLUA_METAMETHOD	proto:__call(name, desc)	Creates a Proto object.	
389	Proto	WSLUA_METHOD	proto:register_heuristic(listname, func)	Registers a heuristic dissector function for this Proto protocol, for the given heuristic list name.	
390	Proto	WSLUA_ATTRIBUTE	proto.dissector	RW The protocol's dissector, a function you define.	
391	Proto	WSLUA_ATTRIBUTE	proto.prefs	RO The preferences of this dissector.	
392	Proto	WSLUA_ATTRIBUTE	proto.prefs_changed	WO The preferences changed routine of this dissector, a Lua function you define.	
393	Proto	WSLUA_ATTRIBUTE	proto.init	WO The init routine of this dissector, a function you define.	
394	Proto	WSLUA_ATTRIBUTE	proto.name	RO The name given to this dissector.	
395	Proto	WSLUA_ATTRIBUTE	proto.description	RO The description given to this dissector.	
396	Proto	WSLUA_ATTRIBUTE	proto.fields	RW The ProtoField's Lua table of this dissector.	
397	Proto	WSLUA_ATTRIBUTE	proto.experts	RW The expert info Lua table of this `Proto`.	
398	Proto	WSLUA_METAMETHOD	Proto__gc		
399	Proto	WSLUA_METAMETHOD	Proto__tostring	lua_pushfstring(L, "Proto: %s", proto->name);	**Add to documentation ???**
400					
401	Global (Proto)	WSLUA_FUNCTION	register_postdissector(proto, [allfields])	Make a Proto protocol (with a dissector function) a post-dissector. It will be called for every frame after dissection.	/* WSLUA_MODULE Proto Functions For New Protocols A
402	Global (Proto)	WSLUA_FUNCTION	dissect_tcp_pdus(tvb, tree, min_header_size, get_len_func, dissect_func, [desegment])	Make the TCP-layer invoke the given Lua dissection function for each PDU in the TCP segment, of the length returned by the given get_len_func function.	
403					
404	**Struct**	**WSLUA_CLASS_DEFINE_BASE**	**Struct**	**The Struct class offers basic facilities to convert Lua values to and from C-style structs in binary Lua strings.**	**/* WSLUA_MODULE Struct Binary encode/decode supp**
405	Struct	WSLUA_CONSTRUCTOR	Struct.pack(format, value)	Returns a string containing the values arg1, arg2, etc. packed/encoded according to the format string.	
406	Struct	WSLUA_CONSTRUCTOR	Struct.unpack(format, struct, [begin])	Unpacks/decodes multiple Lua values from a given struct-like binary Lua string.	
407	Struct	WSLUA_CONSTRUCTOR	Struct.size(format)	Returns the length of a binary string that would be consumed/handled by the given format string.	
408	Struct	WSLUA_CONSTRUCTOR	Struct.values(format)	Returns the number of Lua values contained in the given format string.	
409	Struct	WSLUA_CONSTRUCTOR	Struct.tohex(bytestring, [lowercase], [separator])	Converts the passed-in binary string to a hex-ascii string.	
410	Struct	WSLUA_CONSTRUCTOR	Struct.fromhex(hexbytes, [separator])	Converts the passed-in hex-ascii string to a binary string.	
411	Struct	WSLUA_METAMETHOD	Struct__gc		
412					
413	**TreeItem**	**WSLUA_CLASS_DEFINE**	**TreeItem**	**TreeItems represent information in the packet details pane of Wireshark, and the packet details view of TShark.**	**/* WSLUA_MODULE Tree Adding Information To The Di**
414	TreeItem	WSLUA_METHOD	treeitem:add_packet_field(protofield, [tvbrange], encoding, [label])	Adds a new child tree for the given ProtoField object to this tree item, returning the new child TreeItem.	
415	TreeItem	WSLUA_METHOD	treeitem:add([protofield], [tvbrange], [value], [label])	Adds a child item to this tree item, returning the new child TreeItem. (Big Endian)	
416	TreeItem	WSLUA_METHOD	treeitem:add_le([protofield], [tvbrange], [value], [label])	Adds a child item to this tree item, returning the new child TreeItem. (Little Endian)	
417	TreeItem	WSLUA_ATTRIBUTE	treeitem.text	RW Set/get the TreeItem's display string (string).	
418	TreeItem	WSLUA_METHOD	treeitem:set_text(text)	Sets the text of the label.	
419	TreeItem	WSLUA_METHOD	treeitem:append_text(text)	Appends text to the label.	
420	TreeItem	WSLUA_METHOD	treeitem:prepend_text(text)	Prepends text to the label.	
421	TreeItem	WSLUA_METHOD	treeitem:add_expert_info([group], [severity], [text])	Sets the expert flags of the item and adds expert info to the packet.	
422	TreeItem	WSLUA_METHOD	treeitem:add_proto_expert_info(expert, [text])	Sets the expert flags of the tree item and adds expert info to the packet.	
423	TreeItem	WSLUA_METHOD	treeitem:add_tvb_expert_info(expert, tvb, [text])	Sets the expert flags of the tree item and adds expert info to the packet associated with the Tvb or TvbRange bytes in the packet.	
424	TreeItem	WSLUA_ATTRIBUTE	treeitem.visible	RO Get the TreeItem's subtree visibility status (boolean).	
425	TreeItem	WSLUA_ATTRIBUTE	treeitem.generated	RW Set/get the TreeItem's generated state (boolean).	
426	TreeItem	WSLUA_METHOD	treeitem:set_generated([bool])	Marks the TreeItem as a generated field (with data inferred but not contained in the packet).	
427	TreeItem	WSLUA_ATTRIBUTE	treeitem.hidden	RW Set/get TreeItem's hidden state (boolean).	
428	TreeItem	WSLUA_METHOD	treeitem:set_hidden([bool])	Marks the TreeItem as a hidden field (neither displayed nor used in filters). Deprecated	
429	TreeItem	WSLUA_ATTRIBUTE	treeitem.len	RW Set/get TreeItem's length inside tvb, after it has already been created.	
430	TreeItem	WSLUA_METHOD	treeitem:set_len(len)	Set TreeItem's length inside tvb, after it has already been created.	
431	TreeItem	WSLUA_METHOD	treeitem:referenced(protofield)	Checks if a ProtoField or Dissector is referenced by a filter/tap/UI.	
432	TreeItem	WSLUA_METAMETHOD	treeitem:__tostring()	Returns string debug information about the TreeItem.	

Index	Class	Type	Name	Description	
433	TreeItem	WSLUA_METAMETHOD	TreeItem__gc		
434					
435	**Tvb**	**WSLUA_CLASS_DEFINE**	**Tvb**	**A Tvb represents the packet's buffer.**	**/* WSLUA_MODULE Tvb Functions For Handling Packe**
436	Tvb	WSLUA_METAMETHOD	tvb:__tostring()	Convert the bytes of a Tvb into a string. This is primarily useful for debugging purposes since the string will be truncated if it is too long.	
437	Tvb	WSLUA_METAMETHOD	Tvb__gc		
438	Tvb	WSLUA_METHOD	tvb:reported_len()	Obtain the reported length (length on the network) of a Tvb.	
439	Tvb	WSLUA_METHOD	tvb:captured_len()	Obtain the captured length (amount saved in the capture process) of a Tvb.	
440	Tvb	WSLUA_METHOD	tvb:len()	Obtain the captured length (amount saved in the capture process) of a Tvb. Same as captured_len; kept only for backwards compatibility	
441	Tvb	WSLUA_METHOD	tvb:reported_length_remaining()	Obtain the reported (not captured) length of packet data to end of a Tvb or 0 if the offset is beyond the end of the Tvb.	
442	Tvb	WSLUA_METHOD	tvb:bytes([offset], [length])	Obtain a ByteArray from a Tvb.	
443	Tvb	WSLUA_METHOD	tvb:offset()	Returns the raw offset (from the beginning of the source Tvb) of a sub Tvb.	
444	Tvb	WSLUA_METAMETHOD	tvb:__call()	Equivalent to tvb:range(…)	
445	Tvb	WSLUA_METHOD	tvb:range([offset], [length])	Creates a TvbRange from this Tvb.	
446	Tvb	WSLUA_METHOD	tvb:raw([offset], [length])	Obtain a Lua string of the binary bytes in a Tvb.	
447	Tvb	WSLUA_METAMETHOD	tvb:__eq()	Checks whether contents of two Tvbs are equal.	
448					
449	**TvbRange**	**WSLUA_CLASS_DEFINE**	**TvbRange**	**A TvbRange represents a usable range of a Tvb and is used to extract data from the Tvb that generated it.**	**/* WSLUA_MODULE Tvb Functions For Handling Packe**
450	TvbRange	WSLUA_METHOD	tvbrange:tvb()	Creates a new Tvb from a TvbRange.	
451	TvbRange	WSLUA_METHOD	tvbrange:uint()	Get a Big Endian (network order) unsigned integer from a TvbRange.	
452	TvbRange	WSLUA_METHOD	tvbrange:le_uint()	Get a Little Endian unsigned integer from a TvbRange.	
453	TvbRange	WSLUA_METHOD	tvbrange:uint64()	Get a Big Endian (network order) unsigned 64 bit integer from a TvbRange, as a UInt64 object.	
454	TvbRange	WSLUA_METHOD	tvbrange:le_uint64()	Get a Little Endian unsigned 64 bit integer from a TvbRange, as a UInt64 object.	
455	TvbRange	WSLUA_METHOD	tvbrange:int()	Get a Big Endian (network order) signed integer from a TvbRange.	
456	TvbRange	WSLUA_METHOD	tvbrange:le_int()	Get a Little Endian signed integer from a TvbRange.	
457	TvbRange	WSLUA_METHOD	tvbrange:int64()	Get a Big Endian (network order) signed 64 bit integer from a TvbRange, as an Int64 object.	
458	TvbRange	WSLUA_METHOD	tvbrange:le_int64()	Get a Little Endian signed 64 bit integer from a TvbRange, as an Int64 object.	
459	TvbRange	WSLUA_METHOD	tvbrange:float()	Get a Big Endian (network order) floating point number from a TvbRange.	
460	TvbRange	WSLUA_METHOD	tvbrange:le_float()	Get a Little Endian floating point number from a TvbRange.	
461	TvbRange	WSLUA_METHOD	tvbrange:ipv4()	Get an IPv4 Address from a TvbRange, as an Address object.	
462	TvbRange	WSLUA_METHOD	tvbrange:le_ipv4()	Get an Little Endian IPv4 Address from a TvbRange, as an Address object.	
463	TvbRange	WSLUA_METHOD	tvbrange:ipv6()	Get an IPv6 Address from a TvbRange, as an Address object.	
464	TvbRange	WSLUA_METHOD	tvbrange:ether()	Get an Ethernet Address from a TvbRange, as an Address object.	
465	TvbRange	WSLUA_METHOD	tvbrange:nstime([encoding])	Obtain a time_t structure from a TvbRange, as an NSTime object.	
466	TvbRange	WSLUA_METHOD	tvbrange:le_nstime()	Obtain a nstime from a TvbRange, as an NSTime object.	
467	TvbRange	WSLUA_METHOD	tvbrange:string([encoding])	Obtain a string from a TvbRange.	
468	TvbRange	WSLUA_METHOD	tvbrange:ustring()	Obtain a Big Endian (network order) UTF-16 encoded string from a TvbRange.	
469	TvbRange	WSLUA_METHOD	tvbrange:le_ustring()	Obtain a Little Endian UTF-16 encoded string from a TvbRange.	
470	TvbRange	WSLUA_METHOD	tvbrange:stringz([encoding])	Obtain a zero terminated string from a TvbRange.	
471	TvbRange	WSLUA_METHOD	tvbrange:strsize([encoding])	Find the size of a zero terminated string from a TvbRange.	
472	TvbRange	WSLUA_METHOD	tvbrange:ustringz()	Obtain a Big Endian (network order) UTF-16 encoded zero terminated string from a TvbRange.	
473	TvbRange	WSLUA_METHOD	tvbrange:le_ustringz()	Obtain a Little Endian UTF-16 encoded zero terminated string from a TvbRange	
474	TvbRange	WSLUA_METHOD	tvbrange:bytes([encoding])	Obtain a ByteArray from a TvbRange.	
475	TvbRange	WSLUA_METHOD	tvbrange:bitfield([position], [length])	Get a bitfield from a TvbRange.	
476	TvbRange	WSLUA_METHOD	tvbrange:range([offset], [length])	Creates a sub-TvbRange from this TvbRange.	
477	TvbRange	WSLUA_METHOD	tvbrange:uncompress(name)	Obtain an uncompressed TvbRange from a TvbRange	
478	TvbRange	WSLUA_METAMETHOD	TvbRange__gc		
479	TvbRange	WSLUA_METHOD	tvbrange:len()	Obtain the length of a TvbRange.	
480	TvbRange	WSLUA_METHOD	tvbrange:offset()	Obtain the offset in a TvbRange.	
481	TvbRange	WSLUA_METHOD	tvbrange:raw([offset], [length])	Obtain a Lua string of the binary bytes in a TvbRange.	
482	TvbRange	WSLUA_METAMETHOD	tvbrange:__eq()	Checks whether the contents of two TvbRanges are equal.	
483	TvbRange	WSLUA_METAMETHOD	tvbrange:__tostring()	Converts the TvbRange into a string. The string can be truncated, ...	
484	TvbRange	WSLUA_METAMETHOD	WSLUA_CLASS_MTREG(wslua,concat),	__concat - Concatenation. Invoked similar to addition, using the '..' operator.	
485	TvbRange	WSLUA_METAMETHOD	{"__call", TvbRange_range},	Creates a sub-TvbRange from this TvbRange.	
486					
487	Global (Utility)	WSLUA_FUNCTION	get_version()	Gets the Wireshark version as a string.	/* WSLUA_MODULE Utility Utility Functions */
488	Global (Utility)	WSLUA_FUNCTION	set_plugin_info(table)	Set a Lua table with meta-data about the plugin, such as version.	
489	Global (Utility)	WSLUA_FUNCTION	format_date(timestamp)	Formats an absolute timestamp into a human readable date.	
490	Global (Utility)	WSLUA_FUNCTION	format_time(timestamp)	Formats a relative timestamp in a human readable time.	
491	Global (Utility)	WSLUA_FUNCTION	get_preference(preference)	Get a preference value. @since 3.5.0	
492	Global (Utility)	WSLUA_FUNCTION	set_preference(preference, value)	Set a preference value. @since 3.5.0	

Index	Class	Type	Name	Description	
493	Global (Utility)	WSLUA_FUNCTION	reset_preference(preference)	Reset a preference to default value. @since 3.5.0	
494	Global (Utility)	WSLUA_FUNCTION	apply_preferences()	Write preferences to file and apply changes. @since 3.5.0	
495	Global (Utility)	WSLUA_FUNCTION	report_failure(text)	Reports a failure to the user.	
496	Global (Utility)	WSLUA_FUNCTION	loadfile(filename)	Loads a Lua file and compiles it into a Lua chunk, similar to the standard loadfile but searches additional directories.	
497	Global (Utility)	WSLUA_FUNCTION	dofile(filename)	Loads a Lua file and executes it as a Lua chunk, similar to the standard dofile but searches additional directories.	
498	Global (Utility)	WSLUA_FUNCTION	register_stat_cmd_arg(argument, [action])	Register a function to handle a -z option	
499					
500	Global (Wtap)	WSLUA_FUNCTION	wtap_file_type_subtype_description(filetype)	Get a string describing a capture file type, given a filetype value for that file type.	/* WSLUA_MODULE Wtap Wtap Functions For Handling C
501	Global (Wtap)	WSLUA_FUNCTION	wtap_file_type_subtype_name(filetype)	Get a string giving the name for a capture file type, given a filetype value for that file type.	
502	Global (Wtap)	WSLUA_FUNCTION	wtap_name_to_file_type_subtype(name)	Get a filetype value for a file type, given the name for that file type.	
503	Global (Wtap)	WSLUA_FUNCTION	wtap_pcap_file_type_subtype()	Get the filetype value for pcap files.	
504	Global (Wtap)	WSLUA_FUNCTION	wtap_pcap_nsec_file_type_subtype()	Get the filetype value for nanosecond-resolution pcap files.	
505	Global (Wtap)	WSLUA_FUNCTION	wtap_pcapng_file_type_subtype()	Get the filetype value for pcapng files.	
506					
507	**Int64**	**WSLUA_CLASS_DEFINE_BASE**	**Int64**	**Int64 represents a 64 bit signed integer.**	**WSLUA_MODULE Int64 Handling 64-bit Integers**
508	Int64	WSLUA_METHOD	int64:encode([endian])	Encodes the Int64 number into an 8-byte Lua string using the given endianness.	
509	Int64	WSLUA_CONSTRUCTOR	Int64.decode(string, [endian])	Decodes an 8-byte Lua string, using the given endianness, into a new Int64 object.	
510	Int64	WSLUA_CONSTRUCTOR	Int64.new([value], [highvalue])	Creates a Int64 Object.	
511	Int64	WSLUA_METAMETHOD	int64:__call()	Creates a Int64 object.	
512	Int64	WSLUA_CONSTRUCTOR	Int64.max()	Creates an Int64 of the maximum possible positive value. (9,223,372,036,854,775,807)	
513	Int64	WSLUA_CONSTRUCTOR	Int64.min()	Creates an Int64 of the minimum possible negative value. (-9,223,372,036,854,775,808)	
514	Int64	WSLUA_METHOD	int64:tonumber()	Returns a Lua number of the Int64 value. Note that this may lose precision.	
515	Int64	WSLUA_CONSTRUCTOR	Int64.fromhex(hex)	Creates an Int64 object from the given hexadecimal string.	
516	Int64	WSLUA_METHOD	int64:tohex([numbytes])	Returns a hexadecimal string of the Int64 value.	
517	Int64	WSLUA_METHOD	int64:higher()	Returns a Lua number of the higher 32 bits of the Int64 value. (Could be negative - see wsluarm)	
518	Int64	WSLUA_METHOD	int64:lower()	Returns a Lua number of the lower 32 bits of the Int64 value. This will always be positive.	
519	Int64	WSLUA_METAMETHOD	int64:__tostring()	Converts the Int64 into a string of decimal digits.	
520	Int64	WSLUA_METAMETHOD	int64:__unm()	Returns the negative of the Int64 as a new Int64.	
521	Int64	WSLUA_METAMETHOD	int64:__add()	Adds two Int64 together and returns a new one. The value may wrapped.	
522	Int64	WSLUA_METAMETHOD	int64:__sub()	Subtracts two Int64 and returns a new one. The value may wrapped.	
523	Int64	WSLUA_METAMETHOD	int64:__mul()	Multiplies two Int64 and returns a new one. The value may truncated.	
524	Int64	WSLUA_METAMETHOD	int64:__div()	Divides two Int64 and returns a new one. Integer divide, no remainder.	
525	Int64	WSLUA_METAMETHOD	int64:__mod()	Divides two Int64 and returns a new one of the remainder.	
526	Int64	WSLUA_METAMETHOD	int64:__pow()	The first Int64 is taken to the power of the second Int64, returning a new one.	
527	Int64	WSLUA_METAMETHOD	int64:__eq()	Returns true if both Int64 are equal.	
528	Int64	WSLUA_METAMETHOD	int64:__lt()	Returns true if first Int64 is less than the second.	
529	Int64	WSLUA_METAMETHOD	int64:__le()	Returns true if the first Int64 is less than or equal to the second.	
530	Int64	WSLUA_METAMETHOD	int64:bnot()	Returns a Int64 of the bitwise 'not' operation.	
531	Int64	WSLUA_METAMETHOD	int64:band()	Returns a Int64 of the bitwise 'and' operation with the given number/Int64/UInt64.	
532	Int64	WSLUA_METAMETHOD	int64:bor()	Returns a Int64 of the bitwise 'or' operation, with the given number/Int64/UInt64.	
533	Int64	WSLUA_METAMETHOD	int64:bxor()	Returns a Int64 of the bitwise 'xor' operation, with the given number/Int64/UInt64.	
534	Int64	WSLUA_METAMETHOD	int64:lshift(numbits)	Returns a Int64 of the bitwise logical left-shift operation, by the given number of bits.	
535	Int64	WSLUA_METAMETHOD	int64:rshift(numbits)	Returns a Int64 of the bitwise logical right-shift operation, by the given number of bits.	
536	Int64	WSLUA_METAMETHOD	int64:arshift(numbits)	Returns a Int64 of the bitwise arithmetic right-shift operation, by the given number of bits.	
537	Int64	WSLUA_METAMETHOD	int64:rol(numbits)	Returns a Int64 of the bitwise left rotation operation, by the given number of bits (up to 63).	
538	Int64	WSLUA_METAMETHOD	int64:ror(numbits)	Returns a Int64 of the bitwise right rotation operation, by the given number of bits (up to 63).	
539	Int64	WSLUA_METAMETHOD	int64:bswap()	Returns a Int64 of the bytes swapped. This can be used to convert little-endian 64-bit numbers to big-endian 64 bit numbers or vice versa.	
540	Int64	WSLUA_METAMETHOD	Int64__gc		
541	Int64	WSLUA_METAMETHOD	WSLUA_CLASS_MTREG(wslua,concat),	__concat - Concatenation. Invoked similar to addition, using the '..' operator.	
542					
543	**UInt64**	**WSLUA_CLASS_DEFINE_BASE**	**UInt64**	**UInt64 represents a 64 bit unsigned integer, similar to Int64.**	**WSLUA_MODULE Int64 Handling 64-bit Integers**
544	UInt64	WSLUA_METHOD	uint64:encode([endian])	Encodes the UInt64 number into an 8-byte Lua binary string, using given endianness.	
545	UInt64	WSLUA_CONSTRUCTOR	UInt64.decode(string, [endian])	Decodes an 8-byte Lua binary string, using given endianness, into a new UInt64 object.	
546	UInt64	WSLUA_CONSTRUCTOR	UInt64.new([value], [highvalue])	Creates a UInt64 Object.	
547	UInt64	WSLUA_METAMETHOD	uint64:__call()	Creates a UInt64 object.	
548	UInt64	WSLUA_CONSTRUCTOR	UInt64.max()	Creates a UInt64 of the maximum possible value. (18,446,744,073,709,551,615)	
549	UInt64	WSLUA_CONSTRUCTOR	UInt64.min()	Creates a UInt64 of the minimum possible value. (0)	
550	UInt64	WSLUA_METHOD	uint64:tonumber()	Returns a Lua number of the UInt64 value. This may lose precision.	
551	UInt64	WSLUA_METAMETHOD	uint64:__tostring()	Converts the UInt64 into a string.	
552	UInt64	WSLUA_CONSTRUCTOR	UInt64.fromhex(hex)	Creates a UInt64 object from the given hex string.	
553	UInt64	WSLUA_METHOD	uint64:tohex([numbytes])	Returns a hex string of the UInt64 value.	

Index	Class	Type	Name	Description	
554	UInt64	WSLUA_METHOD	uint64:higher()	Returns a Lua number of the higher 32 bits of the UInt64 value.	
555	UInt64	WSLUA_METHOD	uint64:lower()	Returns a Lua number of the lower 32 bits of the UInt64 value.	
556	UInt64	WSLUA_METAMETHOD	uint64:__unm()	Returns the UInt64 in a new UInt64, since unsigned integers can’t be negated.	
557	UInt64	WSLUA_METAMETHOD	uint64:__add()	Adds two UInt64 together and returns a new one. This may wrap the value.	
558	UInt64	WSLUA_METAMETHOD	uint64:__sub()	Subtracts two UInt64 and returns a new one. This may wrap the value.	
559	UInt64	WSLUA_METAMETHOD	uint64:__mul()	Multiplies two UInt64 and returns a new one. This may truncate the value.	
560	UInt64	WSLUA_METAMETHOD	uint64:__div()	Divides two UInt64 and returns a new one. Integer divide, no remainder.	
561	UInt64	WSLUA_METAMETHOD	uint64:__mod()	Divides two UInt64 and returns a new one of the remainder.	
562	UInt64	WSLUA_METAMETHOD	uint64:__pow()	The first UInt64 is taken to the power of the second UInt64/number, returning a new one.	
563	UInt64	WSLUA_METAMETHOD	uint64:__eq()	Returns true if both UInt64 are equal.	
564	UInt64	WSLUA_METAMETHOD	uint64:__lt()	Returns true if first UInt64 is less than the second.	
565	UInt64	WSLUA_METAMETHOD	uint64:__le()	Returns true if first UInt64 is less than or equal to the second.	
566	UInt64	WSLUA_METHOD	uint64:bnot()	Returns a UInt64 of the bitwise 'not' operation.	
567	UInt64	WSLUA_METHOD	uint64:band()	Returns a UInt64 of the bitwise 'and' operation, with the given number/Int64/UInt64.	
568	UInt64	WSLUA_METHOD	uint64:bor()	Returns a UInt64 of the bitwise 'or' operation, with the given number/Int64/UInt64.	
569	UInt64	WSLUA_METHOD	uint64:bxor()	Returns a UInt64 of the bitwise 'xor' operation, with the given number/Int64/UInt64.	
570	UInt64	WSLUA_METHOD	uint64:lshift(numbits)	Returns a UInt64 of the bitwise logical left-shift operation, by the given number of bits.	
571	UInt64	WSLUA_METHOD	uint64:rshift(numbits)	Returns a UInt64 of the bitwise logical right-shift operation, by the given number of bits.	
572	UInt64	WSLUA_METHOD	uint64:arshift(numbits)	Returns a UInt64 of the bitwise arithmetic right-shift operation, by the given number of bits.	
573	UInt64	WSLUA_METHOD	uint64:rol(numbits)	Returns a UInt64 of the bitwise left rotation operation, by the given number of bits (up to 63).	
574	UInt64	WSLUA_METHOD	uint64:ror(numbits)	Returns a UInt64 of the bitwise right rotation operation, by the given number of bits (up to 63).	
575	UInt64	WSLUA_METHOD	uint64:bswap()	Returns a UInt64 of the bytes swapped. This can be used to convert little-endian 64-bit numbers to big-endian 64 bit numbers or vice versa.	
576	UInt64	WSLUA_METAMETHOD	UInt64__gc		
577	UInt64	WSLUA_METAMETHOD	WSLUA_CLASS_MTREG(wslua,concat),	__concat - Concatenation. Invoked similar to addition, using the '..' operator.	
# Copyright 2022 Chuck Craft <bubbasnmp [AT] gmail.com>			**Version 0.0 - DRAFT**		
#					
# Wireshark - Network traffic analyzer					
# By Gerald Combs <gerald@wireshark.org>					
# Copyright 1998 Gerald Combs					
#					
# SPDX-License-Identifier: GPL-2.0-or-later					
#					