== DEMO1 ==

Before opening a sample packet
==========
- Different info popups by pressing buttons
- Quit button -> asks for confirmation
- Help button -> show how the search works

Open a simple sample packet
==========
- Open files in different formats
- HEX, BIN, ASCII buttons
- Define only necessary protocol information
- Define a field or two
- Defined bytes marked with red
- Saving the protocol tree
- Deleting a field
- Deleting the whole protocol tree

Protocol info
Name: DEMO1
Description: Testing

Words	**Field abbr**	**Type**
They're taking	what	string
the hobbits	who	string
the	article	string
hobbits	noun	string
Isengard	where	string

-Create the fields in different order
-Save the protocol tree
-Choosing a field to be deleted
-Delete a subfield
-Delete a field with subfields
-Delete the whole protocol tree

== DEMO2 ==

```
Protocol info
Name: DEMO2
Description: DEMO2 Test Protocol
Header length given: 15
Payload length from header field number: 1
```

Bytes	Type	Abbr	Name
0	uint8	f0	Message ID
1-2	uint16	f1	Payload length
3-8	bytes	f2	Source info
3-4	uint16	sf2-0	Source port
	-> valuestring: [22]="SSH", [443]="HTTPS"		
	-What if the given valstr is invalid?		
5-8	ipv4	sf2-1	Source IP
9-14	bytes	f3	Destination info
9-10	uint16	sf3-0	Destination port
	-> valuestring: [22]="SSH", [443]="HTTPS"		
11-14	ipv4	sf3-1	Destination IP

```
-What if the user tries to define fields over the given header length?

15-> Payload part

Dissector table (parent): udp.port
 -How the table name is checked from Wireshark?
Dissector ID (port): 55555
 -What happens if the port number is invalid?
 -What if only either one of them is given?
```

== DEMO3 ==

Protocol info
Name: DEMO3
Description: My Test Protocol
Header delimiter: 0d0d
Payload length: all the remaining bytes

Field delimiter: 0d0a
Subfield delimiter: 20

-Field/subfield coloring
-Field/subfield choosing buttons

Field	**Type**	**Abbr**	**Name**
0	uint8	f0	Type
	-> valstr: [0]="Response", [1]="Request"		
	-> bitmask: 0xFF0000		
1	string	f1	Message info
0	string	sf1-0	Method
1	string	sf1-1	Content
2	string	sf1-2	URI
3	string	sf1-3	Protocol version
2	string	f2	Host
->	Payload part		

Dissector table (parent): udp.port
Dissector ID (port): 12345

== DEMO4 ==

```
Protocol info
Name: DEMO4
Description: DEMO4 Parent Protocol
Header length: last field
Payload delimiter: 0d0a

Field delimiter: 3a

Field      Type       Abbr       Name
0          uint16     f0         Port
      valstr:  [194]="IRC", [6679]="IRC DEMO"
      bitmask: 0xFFFF00
1          string     f1         Server
->    Payload part (IRC channels)

Dissector table (parent): tcp.port
Dissector ID (port): 9876
Dissector table (new): demo4.port
Subdissector ID
           Field: 0

-Dissector file must be placed in .wireshark folder,
 not in the plugins folder.
-Line dofile("...") must be added to init.lua
-Show the dissection result before creating the
 dissector for the subprotocol
```

== DEMO5 ==

Protocol info
Name: DEMO5
Description: DEMO5 Subprotocol
Header delimiter: 0d0a
Payload length: the rest

Field delimiter: 2c

Field	Type	Abbr	Name
0	string	ch1	Channel 1
1	string	ch2	Channel 2
2	string	ch3	Channel 3
3	string	ch4	Channel 4

-> No bytes left for payload anymore

Dissector table (parent): demo4.port
Dissector ID (port): 6679