Maestro MQTT support and MQTT to KNX gateway:

MQTT is light weight protocol ideal for Internet of Things (IoT) contexts.
It can support enormous amount of end points at real-time response.
MQTT is based on:
- “Broker” that stores messages on hierarchical topics structure, and on
- “Agents” that publish the messages to the Broker and/or Subscribe to get real time updates of the messages from the Broker.
MQTT is used for single miniature sensors up to gigantic systems as Facebook and Amazon for their web services.
Now MQTT devices can gain from the wide variety of maestro’s: control functions, Logger, Graphs presentation… and from its advanced gateway capabilities to other system such as Modbus and DMX.
In addition, Maestro can serve as 2 way Gateway between MQTT and KNX, The benefits out of this feature are for example:

* The possibility to connect any number of remote KNX sites over IP combining them into one.
* Connecting KNX to SCADA and big data software for further processing and integration with external systems.
* Integrating MQTT IoT devices with KNX.

Maestro can act as Broker, Agent or both at the same time.

MQTT settings on Maestro Designer:

1. On External Devices add a Broker:

and set it’s parameters:
	1. Name and description – free text used for documentation and clarification.
	2. Log level and log Details – settings used for sending Broker events to Maestro Logger.
	3. Address and Port - IP address of the Broker, if internal broker is used – IP should be set to 127.0.0.1 and port to 1883
	4. User and Password – Broker’s credentials (optional).
	5. Client ID – optional.
	6. Topic Prefix – the root prefix for all topics used on this connection to the Broker (optional).
2. On MQTT Client add MQTT client Protocol/Agent:

and set it’s parameters:
	1. Name and Description – free text used for documentation and clarification.
	2. Service – dropdown menu that let you select the Broker for Agents, from the list of Brokers defined on External Devices (as described on previous step).
	3. Topic Prefix – the middle part of all topics used by this client protocol (optional).
	4. Init Expression:
	Can contains Expression Language code. This code will be executed a single time when the controller starts.
	You can use it to declare Static Variables and initialize them.
	5. Connect Expr:
	Can contain Expression Language code. This code will be executed single time when the controller initializes communication with the Broker.
	6. Idle Expression:
	Can contain Expression Language code.
	When the Maestro’s processor is not busy, this code will be executed approximately once every second, You can use it to implement functions such as: delays and cyclic writing of data to the Broker.
	7. Log level and log Details – settings used for sending Agent’s events to Maestro debug Logger.
3. Now you can start adding Subscribers (receiving data from the Broker) and Publishers (sending data to the Broker).


	1. Publishers has the following fields:
		1. Name and Description - free text used for documentation and clarification.
		2. Topic - The last part of the Topic path, The complete Topic path therefore is:
		[Broker Prefix][Client Prefix][ Publisher Topic], and has to specify the exact path of a unique Topic.
		Please notice - and have to including “/”, Maestro Designer will not add or remove any characters.
		3. Expression – can Contain Expression Language code. Any time when Publisher is assigned a value – it will execute its expression language code once.
		The value will be assigned to a variable named *arg*.
		to publish data use the “*return* = data;” command. If the Expression field is left empty – the value assigned to the publisher will be sent directly to the Broker.
		4. Type – the format used for the data sent to the Broker.
		5. Retain – When checked: a published message will be sent immediately to all current subscribers and in addition will be saved, on the Broker, and send immediately to new subscribers.
		When unchecked: only current subscribers will get the new messages and then the message will be discarded from the Broker. New subscribers will get no message on subscription.
		6. QoS – Quality of Service as defined by the MQTT standard.
		7. Log level and log Details – settings used for sending Agent’s events to Maestro Logger.
	2. Subscribers has the following fields:
		1. Name and Description - free text used for documentation and clarification.
		2. Topic Filter - The last part of the topic path. The complete Topic path therefore is:
		[Broker Prefix][Client Prefix][ Subscriber Topic Filter],
		please notice - you have to specify the exact path including “/”, Maestro Designer will not add or remove any characters. A filter make use of MQTT’s Wildcard + # and $. To get messages from more than one publisher.
		3. Expression – Can contain Expression Language code. Any time when message arrives - the code will be executed once.
		The value of the message will be assigned to a variable named *argv[]* according to filter’s field settings.
		If no expression is set - The value of the message will be assigned to the Subscriber.
		4. Filter – can contain a regular expression. The regular expression examine incoming messages, identify parts that match a defined string patterns and pass selected parts to the Expression via *argv[]* variable. If this field is left empty – the complete message will be past to the Subscriber.
		5. Drop: when unchecked - all incoming messages will be passed to the subscriber.
		When checked - only incoming messages different form last messes will be passed to the subscriber (“on change” only).
		6. Log level and log Details – settings used for sending Agent’s events to Maestro Logger.