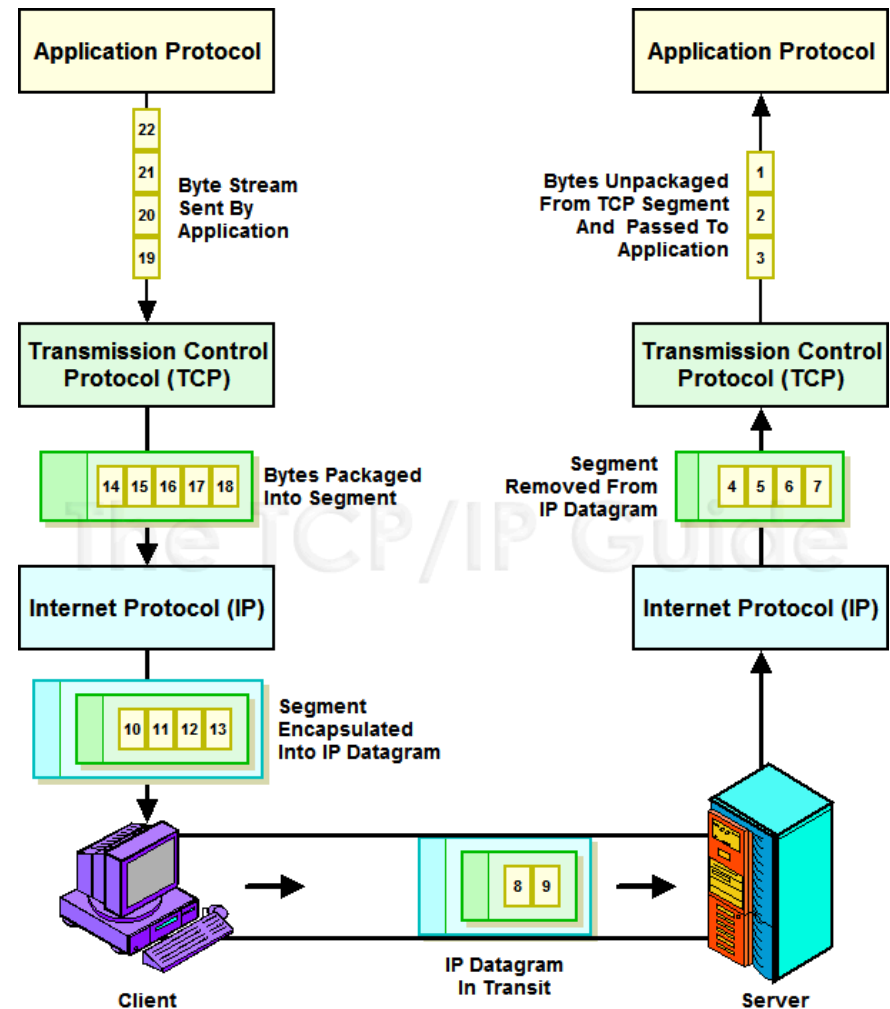


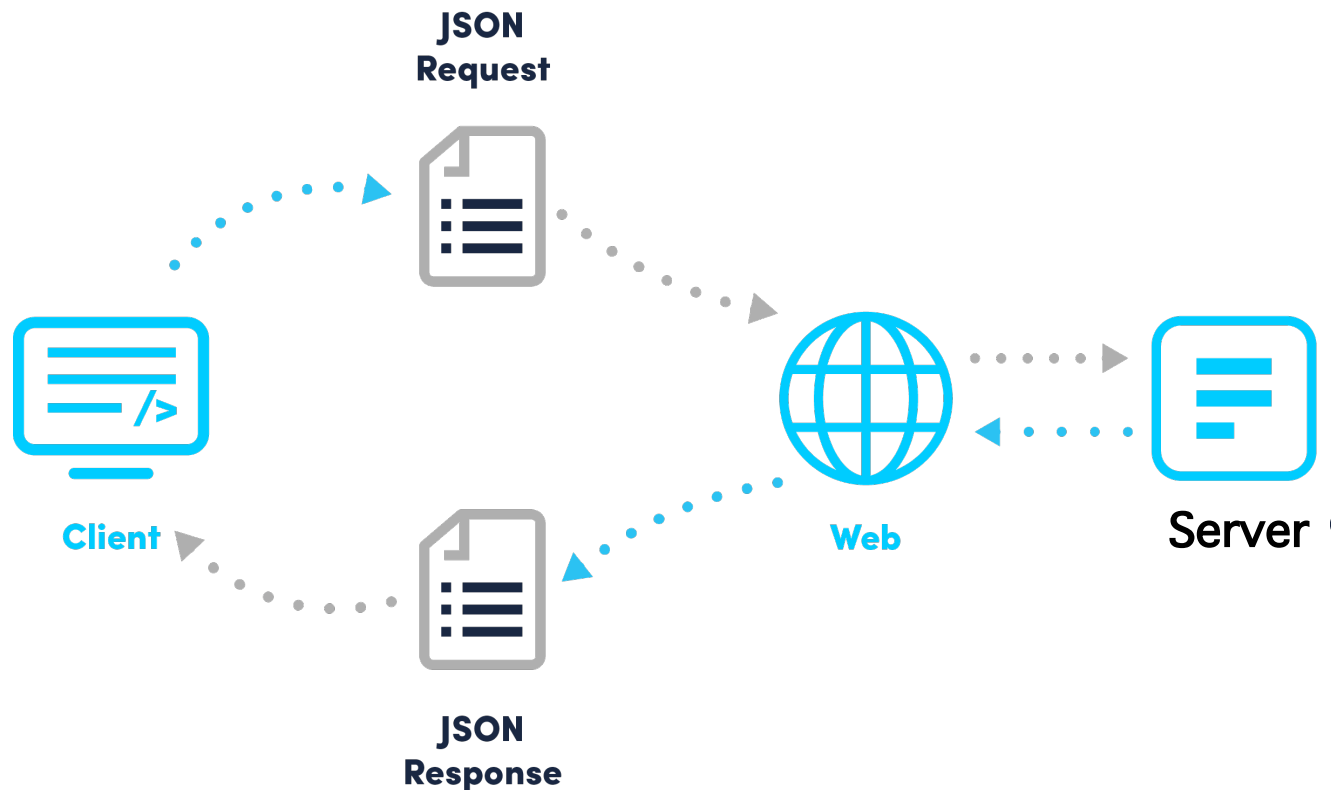
# From “byte streams” to “messages”

- The “old” vision of data communication was based on reliable byte streams, i.e., TCP
- Nowadays messages interchange is becoming more common
  - E.g., Twitter, Whatsapp, Instagram, Snapchat, Facebook,...
- Actually is not that new...
  - emails: SMTP+MIME,
  - FTP,



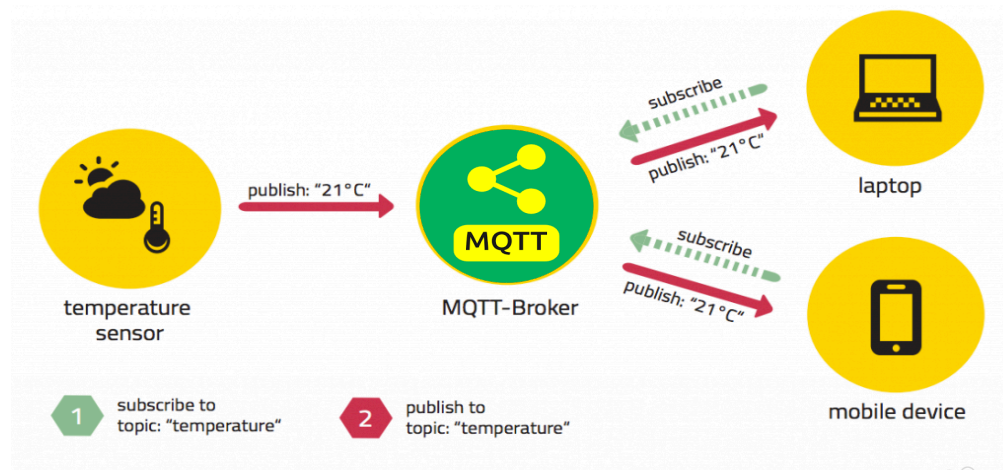
# Request/response approach

- **REST**: Representational State Transfer
- Widely used; based on HTTP
- *Lighter version: **CoAP** (Constrained Application Protocol)*



# Pub/sub approach

- Pub/Sub separate a client, who is sending a message about a specific topic, called publisher, from another client (or more clients), who is receiving the message, called subscriber.
- There is a third component, called broker, which is known by both the publisher and subscriber, which filters all incoming messages and distributes them accordingly.

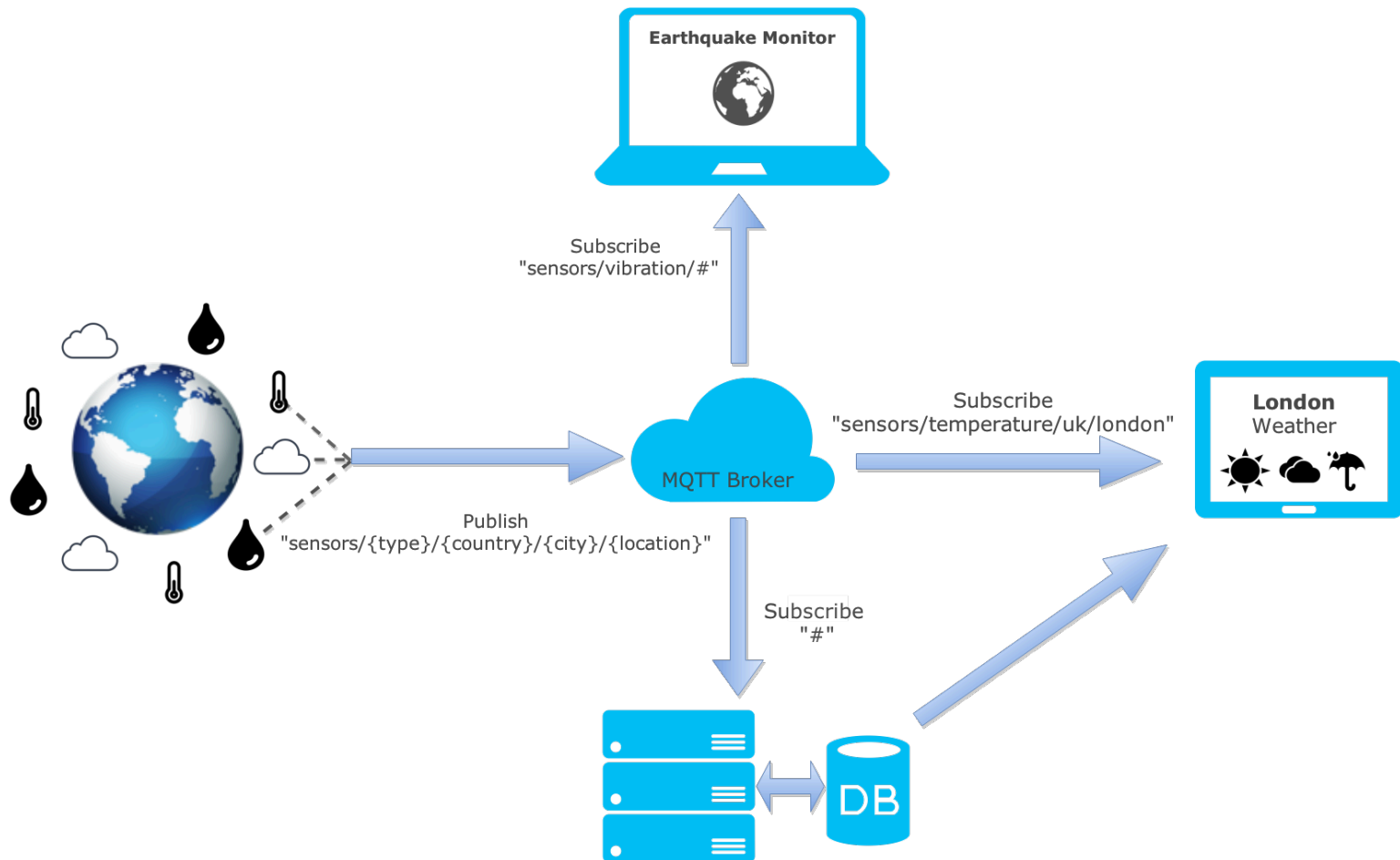


- Various protocols:  
MQTT, AMQP, XMPP (was Jabber)

- Growing technique

E.g., <https://cloud.google.com/iot/docs/how-tos/mqtt-bridge>

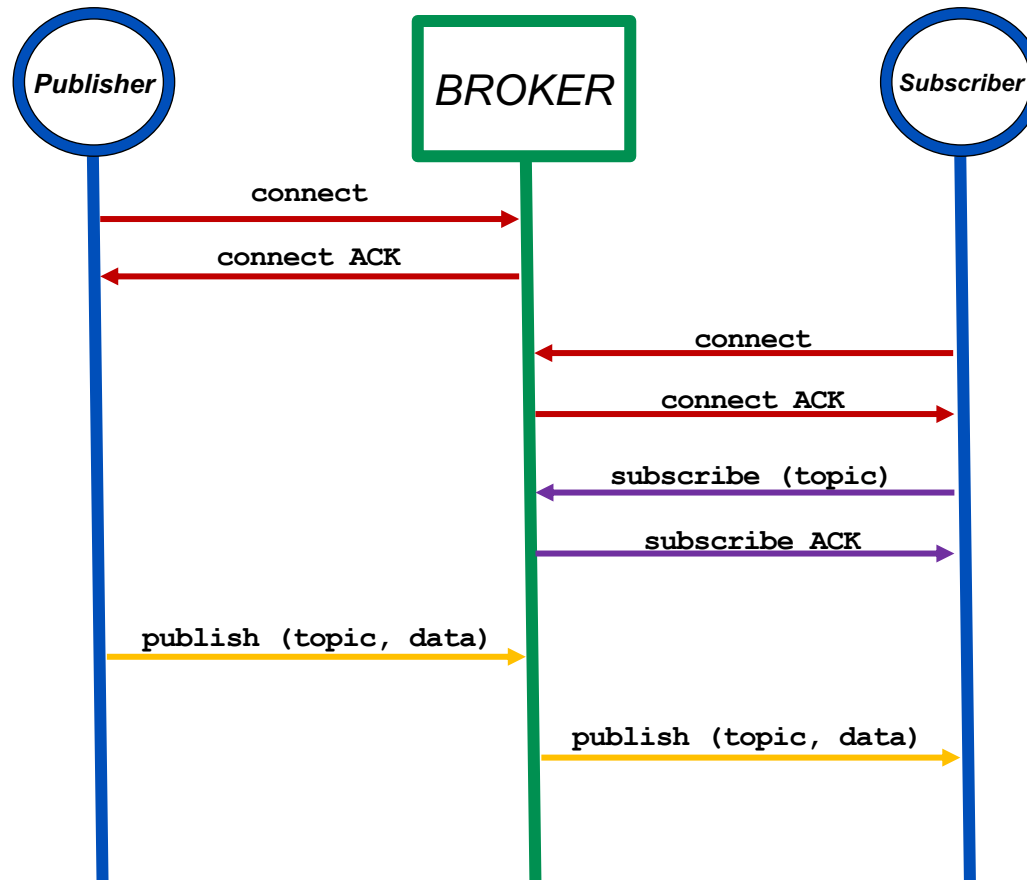
# An example



Source: <https://zoetrope.io/tech-blog/brief-practical-introduction-mqtt-protocol-and-its-application-iot>

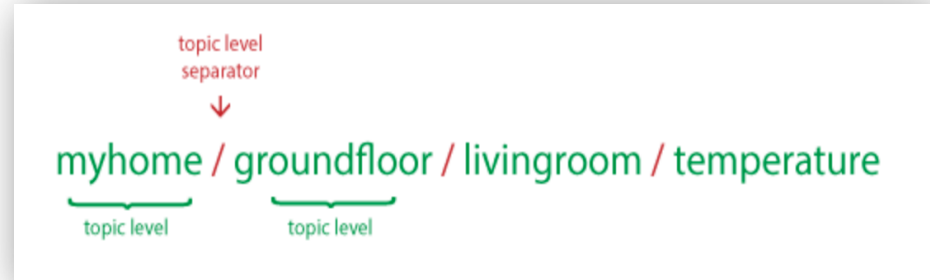
- A lightweight publish-subscribe protocol that can run on embedded devices and mobile platforms → <http://mqtt.org/>
  - Low power usage.
  - Binary compressed headers
  - Maximum message size of 256MB
    - not really designed for sending large amounts of data
    - better at a high volume of low size messages.
- Documentation sources:
  - The MQTT community wiki:
    - <https://github.com/mqtt/mqtt.github.io/wiki>
  - A very good tutorial:
    - <http://www.hivemq.com/mqtt-essentials/>

# Publish/subscribe interactions sequence



# Topics

- MQTT Topics are structured in a hierarchy similar to folders and files in a file system using the forward slash ( / ) as a delimiter.
- Allow to create a user friendly and self descriptive **naming structures**
- Topic names are:
  - Case sensitive
  - use UTF-8 strings.
  - Must consist of at least one character to be valid.
- Except for the \$SYS topic **there is no default or standard topic structure.**



Special \$SYS/ topics

\$SYS/broker/clients/connected  
 \$SYS/broker/clients/disconnected  
 \$SYS/broker/clients/total  
 \$SYS/broker/messages/sent  
 \$SYS/broker/uptime



# Topics wildcards

- Topic subscriptions can have wildcards. These enable nodes to subscribe to groups of topics that don't exist yet, allowing greater flexibility in the network's messaging structure.
  - '+' matches anything at a given tree level
  - '#' matches a whole sub-tree
- Examples:
  - Subscribing to topic `house/#` covers:
    - house/room1/main-light
    - house/room1/alarm
    - house/garage/main-light
    - house/main-door
  - Subscribing to topic `house/+/main-light` covers:
    - house/room1/main-light
    - house/room2/main-light
    - house/garage/main-light
  - but doesn't cover
    - house/room1/side-light
    - house/room2/side-light

# Available MQTT brokers

- The most widely used are:
  - <http://mosquitto.org/>
    - man page: <https://mosquitto.org/man/mosquitto-8.html>
  - <http://www.hivemq.com/>
    - The standard trial version only supports 25 connections.
- And also:
  - <https://www.rabbitmq.com/mqtt.html>
  - <http://activemq.apache.org/mqtt.html>
- A quite complete list can be found here:
  - <https://github.com/mqtt/mqtt.github.io/wiki/servers>

# Cloud based MQTT brokers: CloudMQTT

<https://www.cloudmqtt.com/>

➔ based on Mosquitto

CloudMQTT

Pricing

Documentation

Support

Blog

## Hosted message broker for the Internet of Things



### Power Pug

- Up to 10 000 connections
- No artificial limitations
- Support by e-mail
- Support by phone

**\$ 299**

PER MONTH

Get Now

mized message queues for IoT, ready in seconds.



### Humble Hedgehog

- 25 users/acl rules/connections
- 20 Kbit/s
- 3 bridges
- Support by e-mail

**\$ 5**

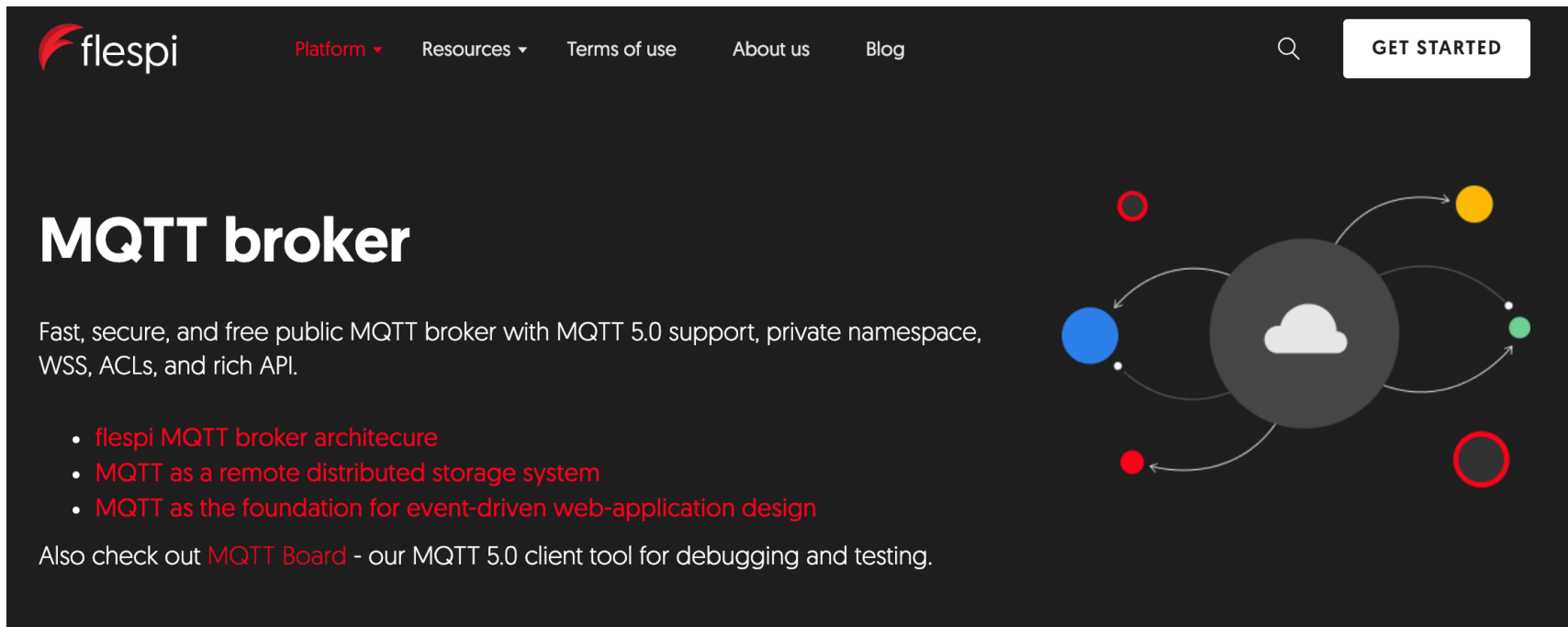
PER MONTH

Get Now



# Cloud based brokers: flespi

<https://flespi.com/mqtt-broker>



The screenshot shows the homepage of the flespi MQTT broker. The header includes the flespi logo, navigation links for Platform, Resources, Terms of use, About us, and Blog, a search icon, and a GET STARTED button. The main heading is 'MQTT broker'. Below it, a description states: 'Fast, secure, and free public MQTT broker with MQTT 5.0 support, private namespace, WSS, ACLs, and rich API.' A list of features includes: flespi MQTT broker architecture, MQTT as a remote distributed storage system, and MQTT as the foundation for event-driven web-application design. A note at the bottom suggests checking out MQTT Board, an MQTT 5.0 client tool for debugging and testing. On the right side, there is a diagram illustrating the MQTT broker architecture, showing a central cloud icon connected to several client devices represented by colored circles (blue, red, yellow, green, and red) with arrows indicating communication flow.

# Open brokers ("Sandboxes")



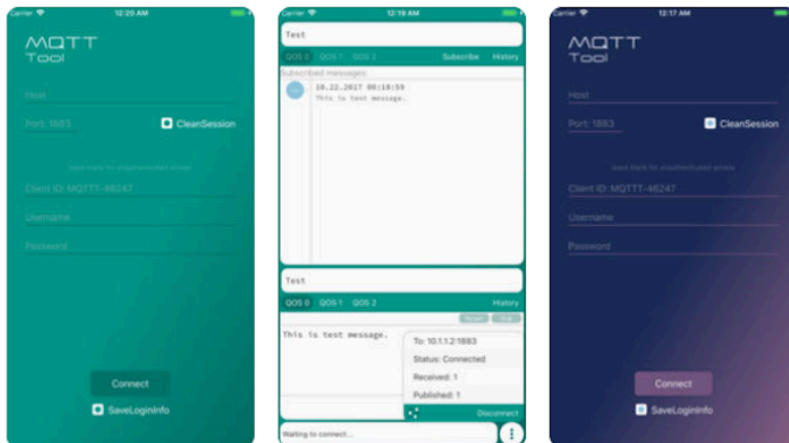
- TCP based:
  - <https://iot.eclipse.org/getting-started/#sandboxes>
    - Hostname: **iot.eclipse.org**
  - <http://test.mosquitto.org/>
    - Hostname: **test.mosquitto.org**
  - <https://www.hivemq.com/mqtt-demo/>
    - Hostname: **broker.hivemq.com**
    - <http://www.mqtt-dashboard.com/>
  - Ports:
    - standard: 1883
    - encrypted: 8883 (*TLS v1.2, v1.1 or v1.0 with x509 certificates*)
  
- Websockets based:
 

○ <a href="http://broker.mqttdashboard.com">broker.mqttdashboard.com</a>	port: 8000
○ <a href="http://test.mosquitto.org">test.mosquitto.org</a>	port: 8080
○ <a href="http://broker.hivemq.com">broker.hivemq.com</a>	port: 8000
  
- [https://github.com/mqtt/mqtt.github.io/wiki/public\\_brokers](https://github.com/mqtt/mqtt.github.io/wiki/public_brokers)

# MQTT clients: iOS











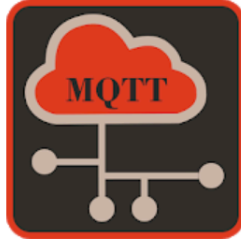

Mqttt  
Utilidades



ICPDAS MQ...  
Utilidades



# MQTT clients: Android

 <p>MQTT Dash (IoT, Smart Home) Routix software</p> <p>★★★★★</p>	 <p>MyMQTT instant:solutions OG</p> <p>★★★★★</p>	 <p>IoT MQTT Panel Rahul Kundu</p> <p>★★★★★</p>	 <p>IoT MQTT Dashboard Nghia TH</p> <p>★★★★★</p>	 <p>MQTT Client Webneurons</p> <p>★★★★★</p>
 <p>MQTT Snooper Maxime Carrier</p> <p>★★★★★</p>	 <p>MQTIZER - Free MQTT Sanyam Arya</p> <p>★★★★★</p>	 <p>Linear MQTT Dashboard ravendmaster</p> <p>★★★★★</p>	 <p>Virtuino MQTT Ilias Lamprou</p> <p>★★★★★</p>	 <p>Mqtt Client Darlei Kroth</p> <p>★★★★★</p>

# MQTT websocket clients

<http://test.mosquitto.org/ws.html>

## MQTT over WebSockets

This is a very early/incomplete/broken example of MQTT over Websockets for test.mosquitto.org. Play around with the buttons below, but don't be surprised if it breaks or isn't very pretty. If you want to develop your own websockets/mqtt app, use the url `ws://test.mosquitto.org/mqtt`, use subprotocol "mqtt" (preferred) or "mqttv3.1" (legacy) and binary data. Then just treat the websocket as a normal socket connection and read/write MQTT packets.

### Usage

Click Connect, then use the Publish and/or Subscribe buttons. You should see text appear below. If you've got another mqtt client available, try subscribe to a topic here then use your other client to send a message to that topic.

Connect Disconnect

Topic:

Payload:

Publish

Topic: \$SYS/#

Subscribe Unsubscribe

### Broker

### Publish

Subscrib



**HIVEMQ**  
ENTERPRISE MQTT BROKER

Websockets Client Showcase

### Connection

Host  Port  ClientID

Username  Password  Keep Alive  Clean Session ☐

Last-Will Topic  Last-Will QoS  Last-Will Retain ☐

Last-Will Message

Publish

Messages

Subscriptions

<http://mitsuruog.github.io/what-mqtt/>

MQTT on Websocket sample

message clear

Connect / Disconnect

MQTT broker on websocket

Address:

Subscribe / Unsubscribe

Topic:

Publish

Topic:

<http://www.hivemq.com/demos/websocket-client/>





# MQTT Explorer

An all-round MQTT client that provides a structured topic overview