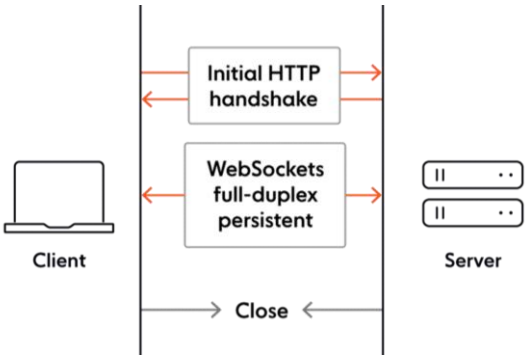


WebSocket (kurz WS) ist ein auf dem Transportprotokoll TCP basierendes Netzwerkprotokoll, das bidirektionale Verbindungen zwischen einer Webanwendung und einem WS-Server herstellt. Diese Verbindungen erlauben einen effizienten, schnellen und dynamischen Informationsaustausch. Sie eignen sich für Echtzeit-Web-Applikationen. Alle derzeit gängigen Web-browser unterstützen WS. Die WS-Verbindung wird in einem Handshake-Verfahren über eine HTTP-Anfrage ausgehandelt und aufgebaut. Im Gegensatz zu einer reinen HTTP-Kommunikation ist keine Anfrage des Clients notwendig, um neue Daten des Servers auszuliefern. Die zugrundeliegende TCP-Verbindung bleibt über den kompletten Zeitraum einer WS-Kommunikation bestehen.

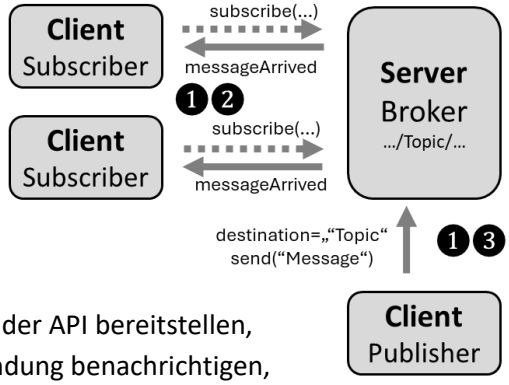
Quelle: https://www.bigdata-insider.de/was-ist-websocket-a-1042523/

Als Programmierschnittstelle wird die Bibliothek „Eclipse Paho“ (mqttws31.min.js) verwendet.

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/paho-mqtt/1.0.1/mqttws31.js"></script>
```

Zu den wesentlichsten unterstützten Funktionen gehören:

1. Verbindung zu und Trennung von einem Server (Identifizierung durch Hostnamen und Port).
2. Abonnieren und Empfangen von Nachrichten von MQTT-Topics.
3. Veröffentlichen von Nachrichten in MQTT-Topics.

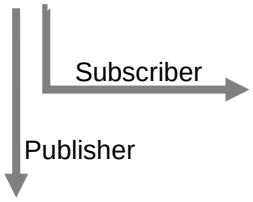


Die API besteht aus zwei Hauptklassen:

Paho.MQTT.Client: enthält Methoden, die die Funktionalität der API bereitstellen, einschließlich der Bereitstellung von Callbacks, die die Anwendung benachrichtigen, wenn eine Nachricht vom Messaging-Server eintrifft oder an diesen zugestellt wird.

Paho.MQTT.Message: Diese kapselt die Nutzlast der Nachricht zusammen mit verschiedenen Attributen, die mit ihrer Zustellung verbunden sind, insbesondere das Ziel, an das sie gesendet wurde (oder gesendet werden soll).

1	①	`client = new Paho.MQTT.Client( hostname , port , clientId );`
	②	`client.onMessageArrived = onMessageArrived; // nur notwendig für Subscriber`
	③	`client.connect({onSuccess:onConnect});`



2	①	`function onConnect() {`
	②	`  client.subscribe((String) "Topic"); }`
	③	`function onMessageArrived(message) {`
	④	`  console.log("empfangen:"+ message.payloadString); }`

3	①	`function onConnect() {`
	②	`  message = new Paho.MQTT.Message( (String) "Nachricht");`
	③	`  message.destinationName = ((String) "Topic");`
	④	`  client.send(message); }`

Die Standard-Ports eines MQTT-Servers (z.B.: test.mosquitto.org) lauten:

1883: Unverschlüsselte MQTT-Verbindungen
8883: Verschlüsselte MQTT-Verbindungen (TLS/SSL)
8080: Unverschlüsselte WebSocket-Verbindungen für MQTT
8881: Verschlüsselte WebSocket-Verbindungen für MQTT (TLS/SSL)