Build your own Open Source IoT Project From A to Z

Javaland 2015, Matthias Zimmermann
Why IoT?
Why Open Source?
Why your Project?
Why IoT?

«The IoT can potentially transform nearly every industry to change the way we live and work, locally and globally. »

www.cisco.com
Why Open Source?

- Faster Innovation
- Better Quality / Security
- Lower Business Risk
- Lower Costs
Why your own IoT Project?

It's fun!

Get your kids excited...

Play with Software AND Hardware

You will learn a lot

Because you can ☺☺☺☺
MQTT Protocol
What is MQTT?

MQTT is a Protocol for the IoT
→ Publish Subscribe
→ Open and Standardized
→ Simple
→ Efficient
→ Robust
MQTT is Open and Standardized

Open

➔ Vendor Neutral, tons of Implementations
➔ ~ 20 Brokers: Mosquitto, Apache ActiveMQ, ...
➔ ~ 60 Clients: Arduino, C/C++, Java, Objective C, ...

Standardized

➔ MQTT v3.1.1 is an OASIS Standard
MQTT is Simple

5 Protocol Verbs
- connect
- publish
- subscribe
- unsubscribe
- disconnect

3 Callbacks
- deliveryComplete
- messageArrived
- connectionLost
MQTT is Efficient

**Broker**
- Can handle many 10k clients

**Client**
- Small footprint (e.g. Java 147KB without JRE)

**Receiving Messages (compared to HTTPS)**
- 100x more messages
- 100x less energy

**Sending Messages**
- 10x more messages
- 10x less energy
MQTT is Robust

MQTT works for Networks with
- Low bandwidth
- High latency
- Unreliable
- High cost per byte

Networks
- Typically TCP
- But also VSAT, GPRS, 2G....
Connect to Broker

- Client A
- Client B
- Client C
- Client D

Broker
Subscribe to Topic «X»
Publish «A» to Topic «X»
Broker forwards «A» to Subscribers of «X»
**MQTT Topics**

**Needed to Publish and Subscribe**

- Publish Message to a Topic
- Subscribe a Topic (-Filter)

**Topic Organization**

- String of one or more UTF-8 chars
- Topic separator “/“ used to separate topic into levels
- Examples
  - `bsiag.com/munich/4thfloor/temperature/last`
  - `/Bundesliga/Game/BER/BAY`
  - `262a1843-589e-4067-a773-03fbe663bc5e`
The MQTT Broker
The MQTT Broker

Raspberry Pi with Mosquitto and a USB WiFi Adapter
Mosquito

Open Source MQTT Broker

- Eclipse IoT Project
- Lightweight
- Written in C
- Executables for Windows, OSX, Linux, and Raspberry Pi

https://www.eclipse.org/mosquitto/
http://mosquitto.org/
Mosquitto on Raspberry Pi

**Step-by-Step**

1. Prepare SD Card
2. Download and Copy Noobs to SD Card
3. Connect Raspberry to Keyboard, Screen, Mouse, Power
4. Configure Raspberry
5. Add USB WiFi dongle and connection
6. Add Mosquitto Broker
7. Reboot frequently 😊
Links that helped

- SD Card

- Noobs

- Add WiFi
  http://weworkweplay.com/play/automatically-connect-a-raspberry-pi-to-a-wifi-network/

- WiFi USB Dongle
  Go for the Edimax EW-7811Un

- Mosquitto
  http://jpmens.net/2013/09/01/installing-mosquitto-on-a-raspberry-pi/
The MQTT Scout Client
The MQTT Scout Client

Scout Mobile Client
The MQTT Scout Client

Scout Web Client
Eclipse Scout

Open Source Application Framework

- Eclipse Technology Project
- Multi Device Support (Desktop, Web, Mobile)
- Client Server Architecture
- Scales well for large Applications
- Simple to learn
- Based on Java / Eclipse
  - 2016 pure Java Framework

https://www.eclipse.org/scout
Scout Architecture

Client Application
- Mobile
- Web
- Desktop
- Scout Client
- Java / Eclipse

Service Tunnel

Server Application
- Scout Server
- Java / Eclipse
- Webserver
Learn more about Scout

Visit the JavaLand Scout booth...
Scout MQTT Client

Step-by-Step
1. Download and Install Eclipse Scout
   https://www.eclipse.org/downloads/
2. Clone the MQTT Scout Github Repo
   https://github.com/BSI-Business-Systems-Integration-AG/mqtt.git
3. Open Scout IDE with empty workspace
4. Import plugins in scout subfolder of cloned project
The MQTT Paho Client
The MQTT Paho Client

Paho Command Line Client
Paho

Open Source MQTT Client Implementations

- Eclipse IoT Project
- Client Libraries in C/C++, Java, JavaScript, Python, C#
- Java Library very simple to learn

https://www.eclipse.org/paho
public class PahoMqttClient {
    static String broker = "tcp://iot.eclipse.org:1883";
    static String clientId = "Paho Client";
    static String topic = "eclipse/scout/arduino/"
    static String content = "RELAYS ON";

    public static void main(String[] args) {
        if (args.length == 2) {
            topic = args[0];
            content = args[1];
        }
        printConfiguration();

        try {
            MemoryPersistence persistence = new MemoryPersistence();
            MqttClient client = new MqttClient(broker, clientId, persistence);
            MqttConnectOptions opts = new MqttConnectOptions();
            MqttMessage message = new MqttMessage(content.getBytes());

            client.connect(opts);
            client.publish(topic, message);
            client.disconnect();
            System.out.println("Connected to broker");
            System.out.println("Published message "+content+" to topic "+topic);
            System.out.println("Disconnected");
            System.exit(0);
        }
        catch (MqttException me) {
            me.printStackTrace();
        }
    }
}
The MQTT Arduino Client
The MQTT Arduino Client

Arduino Uno with the Arduino WiFi Shield
The MQTT Arduino Client

Relay with modified Power Cable
Arduino PubSubClient

Open Source Client Library for the Arduino

- Works out of the box
- Very simple to use
- QOS 1,2 messaging NOT supported

https://github.com/knolleary/pubsclient
http://knolleary.net/arduino-client-for-mqtt/
Simple Arduino Sketch

```cpp
#include <SPI.h>
#include <Ethernet.h>
#include <PubSubClient.h>

// Update these with values suitable for your network.
byte mac[] = { 0xDE, 0xED, 0xBA, 0xFE, 0xFE, 0xED };
byte ip[] = { 172, 16, 0, 100 };
byte server[] = { 172, 16, 0, 2 };

void callback(char* topic, byte* payload, unsigned int length) {
}

EthernetClient ethClient;
PubSubClient client(server, 1883, callback, ethClient);

void setup() {
  Ethernet.begin(mac, ip);
  if (client.connect("arduinoClient")) {
    client.publish("outTopic","hello world");
    client.subscribe("inTopic");
  }
}

void loop() {
  client.loop();
}
```
Arduino MQTT Client

Step-by-Step

1. Download and Install Arduino IDE
   http://arduino.cc/en/Main/Software
2. Clone the MQTT Scout Github Repo
   https://github.com/BSI-Business-Systems-Integration-AG/mqtt.git
3. Open mqttClient.ino Sketch in Arduino IDE
   And fix WiFi Settings/MQTT Broker for you Setup
4. Put WiFi Shield on Arduino Uno and add Electronics
5. Connect Arduino to Computer
6. Upload Script
**The MQTT Arduino Client**

**What it does**

1. Setup Input/Output Pins
2. Finds correct WiFi
3. Connects to WiFi
4. Connects to MQTT Broker and subscribes to `eclipse/scout/arduino`
5. Listens for Commands
   - **RELAY ON** → Switches Lamp On
   - **RELAY OFF** → Switches Lamp Off
   - **LDR GET** → Publishes LDR to `eclipse/scout/arduino/ldr`
Wrap Up
What have we done?

We learned about a specific IoT project setup
- From Mobile Phone to Arduino and Back
- MQTT Protocol
- Mosquitto Broker
- Paho Client Library
- Eclipse Scout Framework
- Arduino IDE with PubSub Library
- Breadboard with Electronics and AC Relay
What have we missed?

1. Many other IoT Protocols
2. Many other Devices
3. IoT and Cloud
4. Security
5. ...
What next?

Get Gadgets

Get the Software

Start to play ...
Thanks !