MDE

TP1 – Integration with Real Data

2024 - 2025

TOPICS

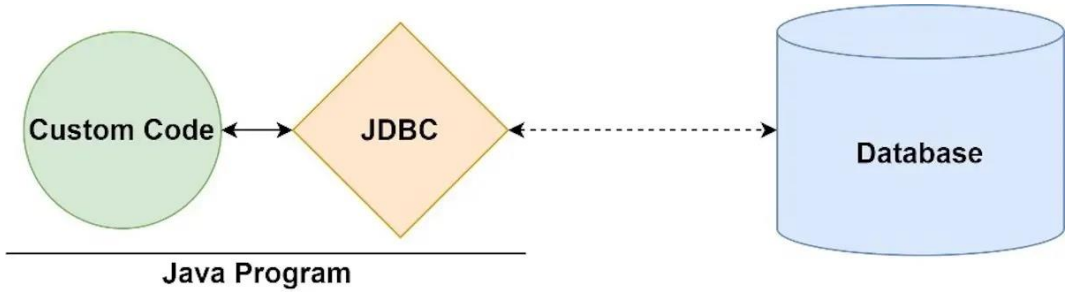
- Integration with our Database
  - JDBC
  - MySQL_Integration library
- Real Data Collection
  - Real System Architecture
  - Data Extraction Protocol (MQTT)
  - Apache Paho
  - MQTT_Library

Integration with Our Database

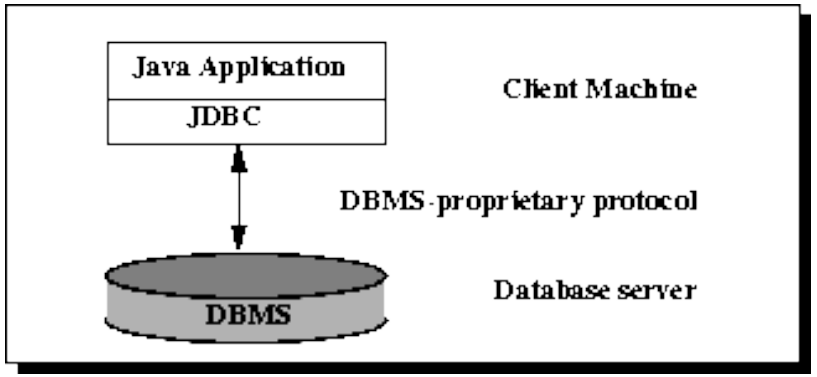
- A Database is commonly accessed by external applications developed in different technologies (JAVA, Python, PHP, JavaScript, etc.)
- In our case, we will integrate our Database with a JAVA Application.
- We will use an API (Application Programming Interface), the JDBC.

- *"JDBC (Java Database Connectivity) is the Java API that manages connecting to a database, issuing queries and commands, and handling result sets obtained from the database."*

https://www.infoworld.com/article/3388036/what-is-jdbc-introduction-to-java-database-connectivity.html



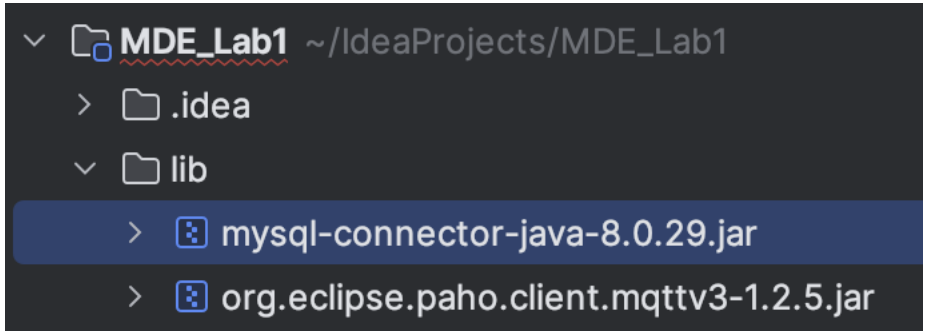
https://www.infoworld.com/article/3388036/what-is-jdbc-introduction-to-java-database-connectivity.html



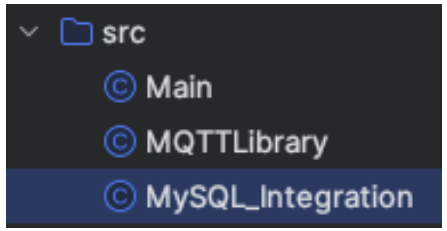
https://docs.oracle.com/javase/tutorial/jdbc/overview/architecture.html

Integration with Our Database

- You can find on the CLIP platform a JAVA project you must use to develop the integration with your database and real data collection.
- The *jar* file required for integrating our database is already included in the project.



- In the project you can find a library (MySQL_Integration) responsible for create/close connection and execution of queries.



- In both files (Main and MySQL_Integration) it was added the following import.

```
import java.sql.*;
```

Integration with Our Database

- In the MySQL_Integration you can find three methods:

```java
public class MySQL_Integration {

    no usages
    static Connection createConnection(String url, String username, String password) throws SQLException {
        return DriverManager.getConnection(url,username,password);
    }

    no usages
    static void closeConnection(Connection conn) throws SQLException {
        conn.close();
    }

    //For SELECT
    no usages
    static ResultSet executeQuery(Connection conn, String query) throws SQLException {
        Statement stmt = conn.createStatement();
        return stmt.executeQuery(query);
    }

    //For INSERT, UPDATE and DELETE
    no usages
    static int executeUpdate(Connection conn, String query) throws SQLException {
        Statement stmt = conn.createStatement();
        return stmt.executeUpdate(query);
    }
}
```

Integration with Our Database

- To use the MySQL_Integration library, you need to specify the following (**Main**):

```
//Database information
String url = "jdbc:mysql://localhost/testDB?useSSL=false";
String username = "testuser";    //Replace with your database username
String password = "test";        //Replace with user password
```

testDB is the name of the database. Replace for the name of your databse

- Then you can use the library as follows (**Main**):

```
try {
    //Start Connection
    Connection conn = MySQL_Integration.createConnection(url,username,password);
    //Execute Query
    ResultSet resultSet = MySQL_Integration.executeQuery(conn, query: "SELECT * FROM customers");
    //Process Result
    while (resultSet.next()) {
        // Process the result set
        System.out.println(resultSet.getString( columnLabel: "id") + "\t" + resultSet.getString( columnLabel: "name"));
    }
    //Close Connection
    MySQL_Integration.closeConnection(conn);
} catch (SQLException e) {
    throw new RuntimeException(e);
}
```

Real System Architecture

Network: MDE

Server with MQTT Broker

Gateway already available

Read

Your JAVA App

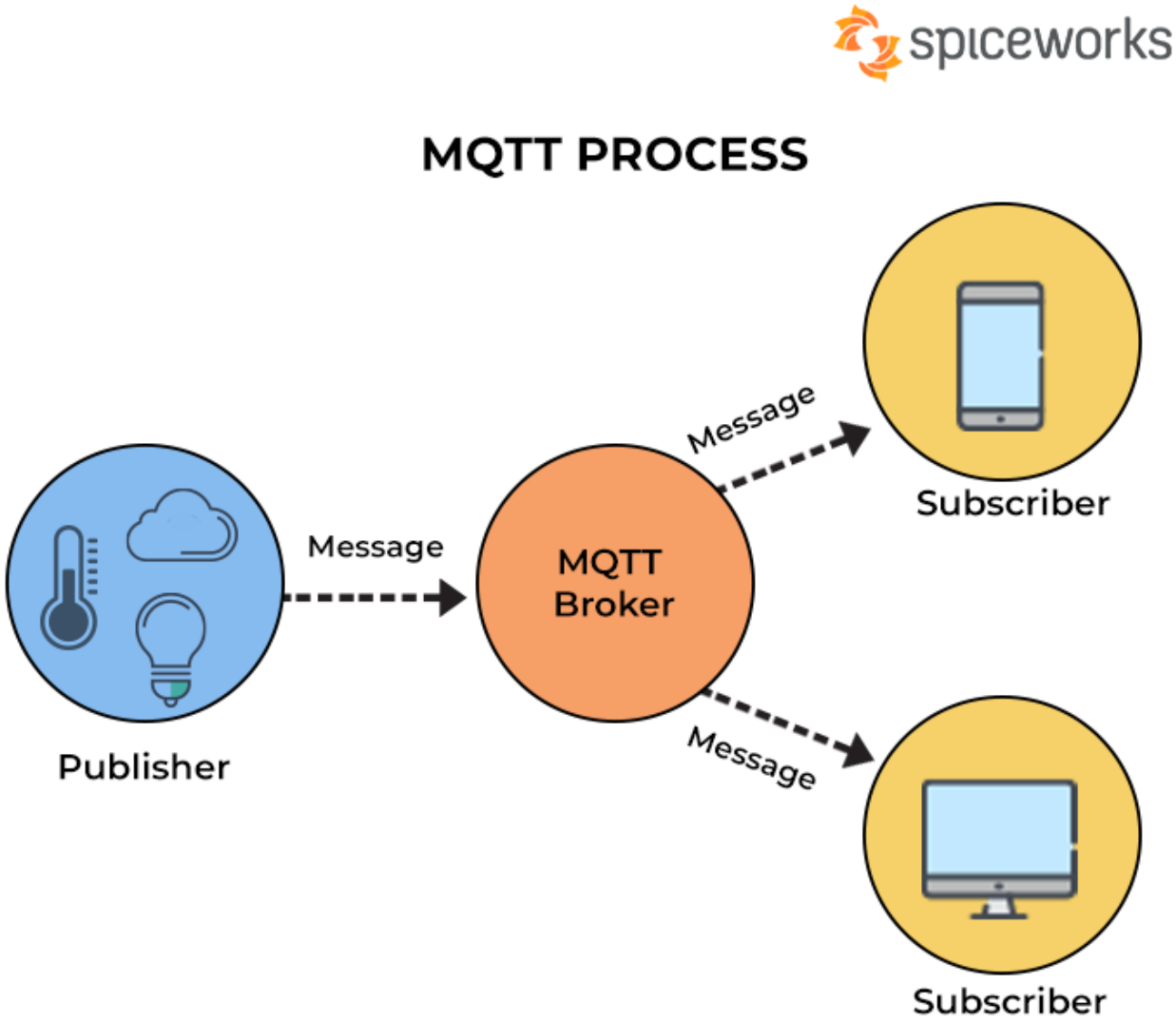
Your Queries

Your Database

These communications are done through a library already available!!!

MQTT

- Devices that generate data:
  - Publishers
- Devices that receive data:
  - Subscribers



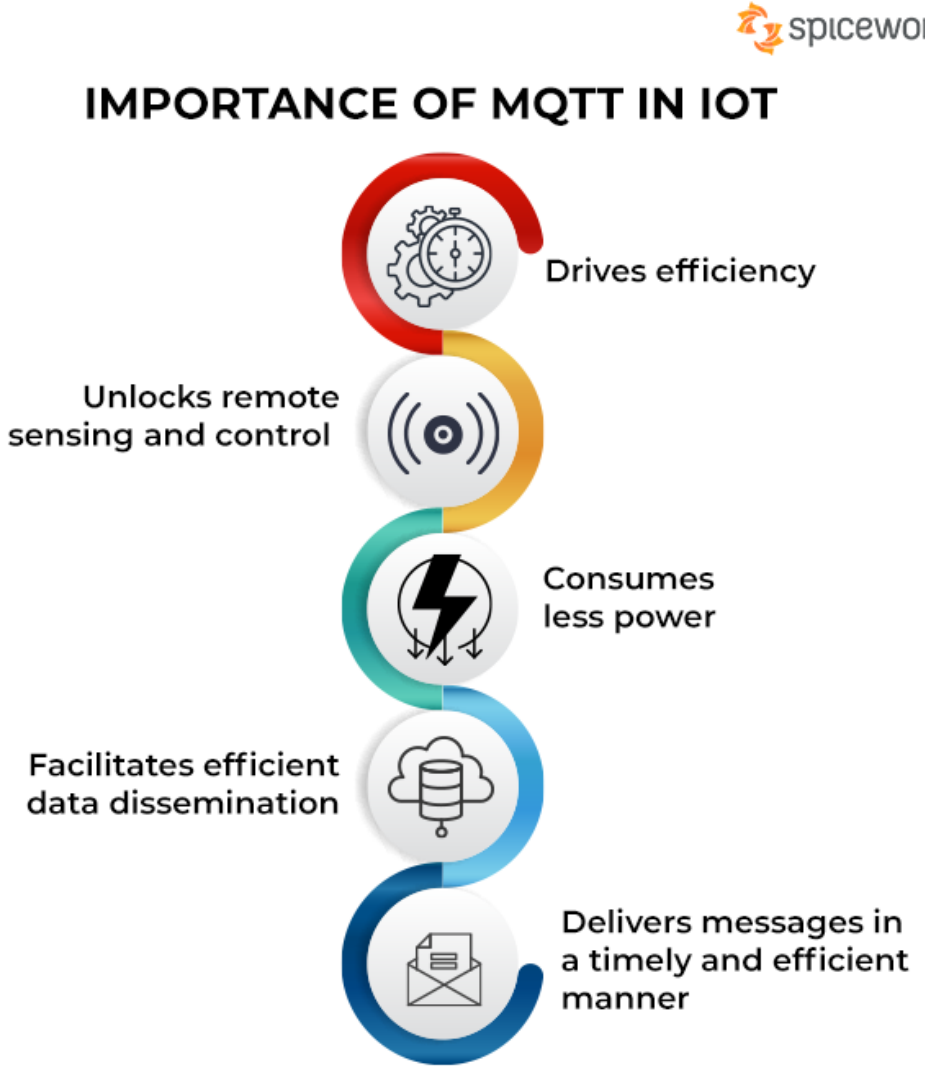
https://www.spiceworks.com/tech/iot/articles/what-is-mqtt/

MQTT

- MQTT is one of the main technologies when implementing IoT (Internet of Things).

- *"Message queuing telemetry transport (MQTT) is a low-bandwidth machine-to-machine protocol for IoT device communication."*

https://www.spiceworks.com/tech/iot/articles/what-is-mqtt/



https://www.spiceworks.com/tech/iot/articles/what-is-mqtt/

MQTT

- Scenarios and sectors where we can use MQTT:
  - Remote Sensing
  - **Smart Cities**
  - Social Media Platforms
  - Home Automation
  - **Smart Farming**
  - Wearables
  - Manufacturing
  - …

https://www.spiceworks.com/tech/iot/articles/what-is-mqtt/

MQTT

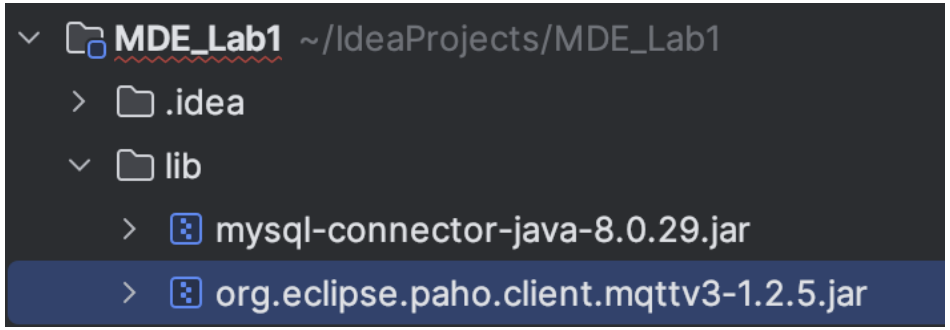
- All the messages are sent/received to/from a topic.
- For example:
  - A device that wants to publish a temperature value can publish its measurement in a topic like *productionID/temperature*.
  - Hence, all the applications that want to receive the values published in this topic must subscribe to the topic *productionID/temperature*.

- Available topics/sensors:
  - production1/temperature
  - production1/humidity
  - production2/temperature
  - production2/humidity

Our JAVA Project for Data Collection

- In order for us to use an MQTT broker to exchange our messages. We need to use an MQTT library in our JAVA Project. The JAVA project that you will use includes the Apache Paho library.

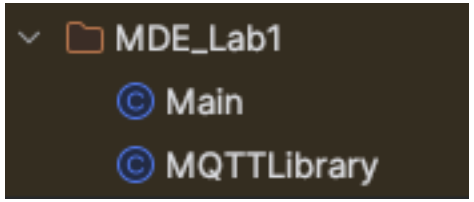
https://eclipse.dev/paho/



- Firstly, we need to be connected to the same network for this to work.
- So, the first thing you should do is connect to the MDE network.
- Network: MDE

Our JAVA Project for Data Collection

- Our JAVA project has a library included to subscribe to the MQTT broker (MQTT Library)



- If you run this project, you will see that the project already collects data from the topics selected:

```
client.connect(connOpts);
client.subscribe( topicFilter: "production1/temperature");
client.subscribe( topicFilter: "production2/temperature");
client.subscribe( topicFilter: "production1/humidity");
client.subscribe( topicFilter: "production2/humidity");
```

- Summary of the available values:
  - 2 temperature sensors
  - 2 humidity sensors

Our JAVA Project for Data Collection

- Whenever a new value is received in the subscribed topics, the following method is triggered (**Main**):

```
public static void messageReceived(String topic, MqttMessage message) {
    System.out.println("Message arrived. Topic: " + topic + " Message: " + message.toString());
}
```

- As you can see, if we run the project, we can see the values received from the different topics:

Required Materials / Tools

- One IDE for Java coding (one of these):
  - IntelliJ IDEA (Recommended)
  - Visual Studio Code
  - Apache NetBeans
  - Eclipse
  - …

FINAL REMARKS

- This is the last challenge for you to go to a remarkable work!
- Let’s do it!!