

MyBatis es un framework popular para el mapeo de objetos-relacional (ORM) en Java que permite una integración sencilla con bases de datos relacionales sin requerir el uso de complejas configuraciones de mapeo usando SQL puro. En este artículo, vamos a crear un proyecto básico de “Hola Mundo” usando MyBatis para conectar una base de datos y trabajar con la clase `Persona`.

1. Configuración del Proyecto

Primero, asegurémonos de configurar un proyecto de Java (puede ser Maven o Gradle) y añadir las dependencias de MyBatis y MySQL (u otro controlador de base de datos que prefieras).

Ejemplo de configuración en Maven (pom.xml):

```
<dependencies>
  <!-- MyBatis Dependency -->
  <dependency>
    <groupId>org.mybatis</groupId>
    <artifactId>mybatis</artifactId>
    <version>3.5.9</version>
  </dependency>
  <!-- MySQL Connector Dependency -->
  <dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>8.0.27</version>
  </dependency>
</dependencies>
```

Como se puede ver la configuración es muy sencilla a nivel de dependencias.

2. Creación de la Tabla Persona en la Base de Datos

A continuación, crearemos una tabla simple llamada Persona en la base de datos. Este script SQL es un ejemplo de cómo crearla:

```
CREATE TABLE Persona (  
    id INT PRIMARY KEY AUTO_INCREMENT,  
    nombre VARCHAR(50),  
    edad INT  
);
```

3. Definición de la Clase Persona

En nuestro proyecto, crearemos la clase Persona, que representa el modelo de datos.

```
package com.arquitecturajava.models;  
  
public class Persona {  
    private int id;  
    private String nombre;  
    private int edad;  
  
    // Constructores  
    public Persona() {}  
  
    public Persona(String nombre, int edad) {  
        this.nombre = nombre;  
        this.edad = edad;  
    }  
}
```

```
// Getters y Setters
public int getId() { return id; }
public void setId(int id) { this.id = id; }

public String getNombre() { return nombre; }
public void setNombre(String nombre) { this.nombre = nombre; }

public int getEdad() { return edad; }
public void setEdad(int edad) { this.edad = edad; }
}
```

4. Configuración de MyBatis

Vamos a configurar MyBatis para que sepa cómo manejar nuestra clase Persona.



En el archivo mybatis-config.xml, podemos especificar la configuración básica:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
  <environments default="development">
    <environment id="development">
      <transactionManager type="JDBC"/>
      <dataSource type="POOLED">
        <property name="driver"
value="com.mysql.cj.jdbc.Driver"/>

```

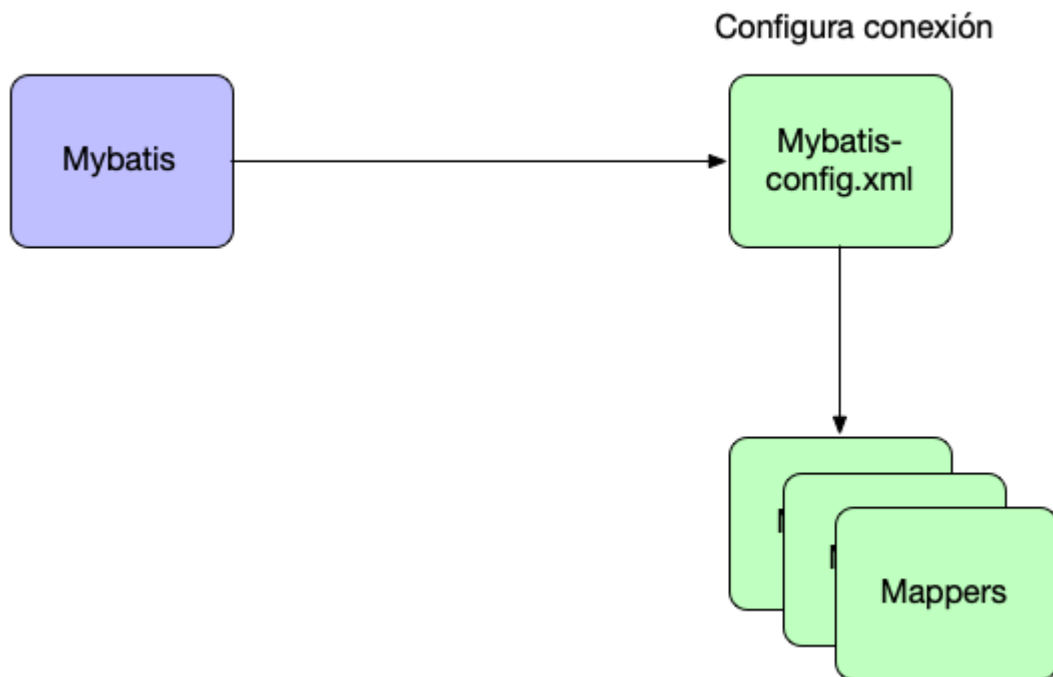
```

        <property name="url"
value="jdbc:mysql://localhost:3306/test"/>
        <property name="username" value="root"/>
        <property name="password" value="password"/>
    </dataSource>
</environment>
</environments>

<mappers>
    <mapper
resource="com/arquitecturajava/mappers/PersonaMapper.xml"/>
</mappers>
</configuration>

```

Junto con ello vemos como referencia a un fichero de Mapeo el fichero de configuración puede tener multiples ficheros de mapeo de consultas:



Veamos su contenido:

```

<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="com.arquitecturajava.mappers.PersonaMapper">
    <insert id="insertarPersona"
parameterType="com.arquitecturajava.models.Persona">
        INSERT INTO Persona (nombre, edad) VALUES (#{nombre},
#{edad});
    </insert>

    <select id="seleccionarPersona" parameterType="int"
resultType="com.arquitecturajava.models.Persona">
        SELECT * FROM Persona WHERE id = #{id};
    </select>

    <select id="seleccionarTodasLasPersonas"
resultType="com.arquitecturajava.models.Persona">
        SELECT * FROM Persona;
    </select>
</mapper>

```

Contiene todas las consultas SQL que se mapearan

5. Creación de un Mapper para Persona

MyBatis usa interfaces y archivos XML para definir los métodos de acceso a la base de datos. Crearemos una interfaz PersonaMapper y su archivo XML de configuración.

```

package com.arquitecturajava.mappers;

import com.arquitecturajava.models.Persona;
import java.util.List;

```

```
public interface PersonaMapper {  
    void insertarPersona(Persona persona);  
    Persona seleccionarPersona(int id);  
    List<Persona> seleccionarTodasLasPersonas();  
}
```

6. Ejecutar una Consulta Sencilla con MyBatis

Ahora podemos escribir una clase Main que configure MyBatis, y luego ejecute una consulta de prueba para insertar y seleccionar datos de la tabla Persona.

```
package com.arquitecturajava;  
  
import com.arquitecturajava.models.Persona;  
import com.arquitecturajava.mappers.PersonaMapper;  
import org.apache.ibatis.io.Resources;  
import org.apache.ibatis.session.SqlSession;  
import org.apache.ibatis.session.SqlSessionFactory;  
import org.apache.ibatis.session.SqlSessionFactoryBuilder;  
  
import java.io.IOException;  
import java.io.Reader;  
  
public class Main {  
    public static void main(String[] args) {  
        try {  
            // Cargar configuración de MyBatis  
            Reader reader = Resources.getResourceAsReader("mybatis-  
config.xml");  
            SqlSessionFactory sqlSessionFactory = new  
SqlSessionFactoryBuilder().build(reader);
```

```

        try (SqlSession session = sqlSessionFactory.openSession())
        {
            PersonaMapper mapper =
session.getMapper(PersonaMapper.class);

            // Insertar una persona
            Persona persona = new Persona("Juan", 30);
            mapper.insertarPersona(persona);
            session.commit(); // Hacer commit a la transacción

            // Seleccionar una persona
            Persona personaSeleccionada =
mapper.seleccionarPersona(persona.getId());
            System.out.println("Persona seleccionada: " +
personaSeleccionada.getNombre() + ", Edad: " +
personaSeleccionada.getEdad());
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
}
}

```

Persona seleccionada: Juan, Edad: 30

Conclusión

Con esto, hemos creado una aplicación básica de “Hola Mundo” usando MyBatis y una clase `Persona`. Esta implementación te permite trabajar con una base de datos relacional de manera eficiente y con mínima configuración adicional, facilitando la conexión y el mapeo de clases Java. A partir de aquí, puedes expandir esta base para manejar más operaciones CRUD y funcionalidades complejas según los requerimientos de tu proyecto.

- [Java JSON con Jackson](#)
- [Java new String y la creación de objetos](#)
- [Java WAR \(el concepto de web archive\)](#)
- [¿Qué es el web.xml?](#)
- [Java Mapping con MapStrut y anotaciones](#)