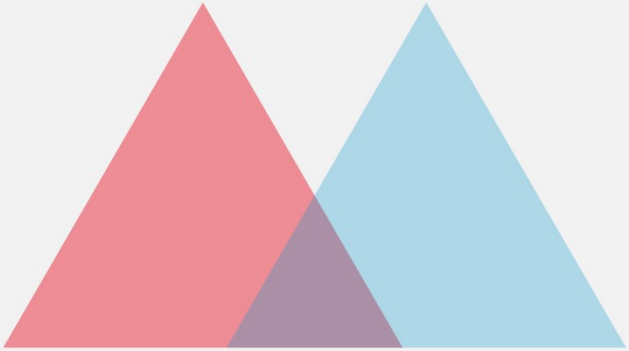
Mybatis



Mybatis란?

Mybatis는 개발자가 지정한 SQL, 저장프로시저 그리고 몇가지 고급 매핑을 지원하는 퍼시스턴스 프레임워크이다. Mybatis는 JDBC로 처리하는 상당부분의 코드와 파라미터 설정및 결과 매핑을 대신해준다. Mybatis는 데이터베이스 레코드에 원시타입과 Map 인터페이스 그리고 자바 POJO 를 설정해서 매핑하기 위해 XML과 애노테이션을 사용할 수 있다.

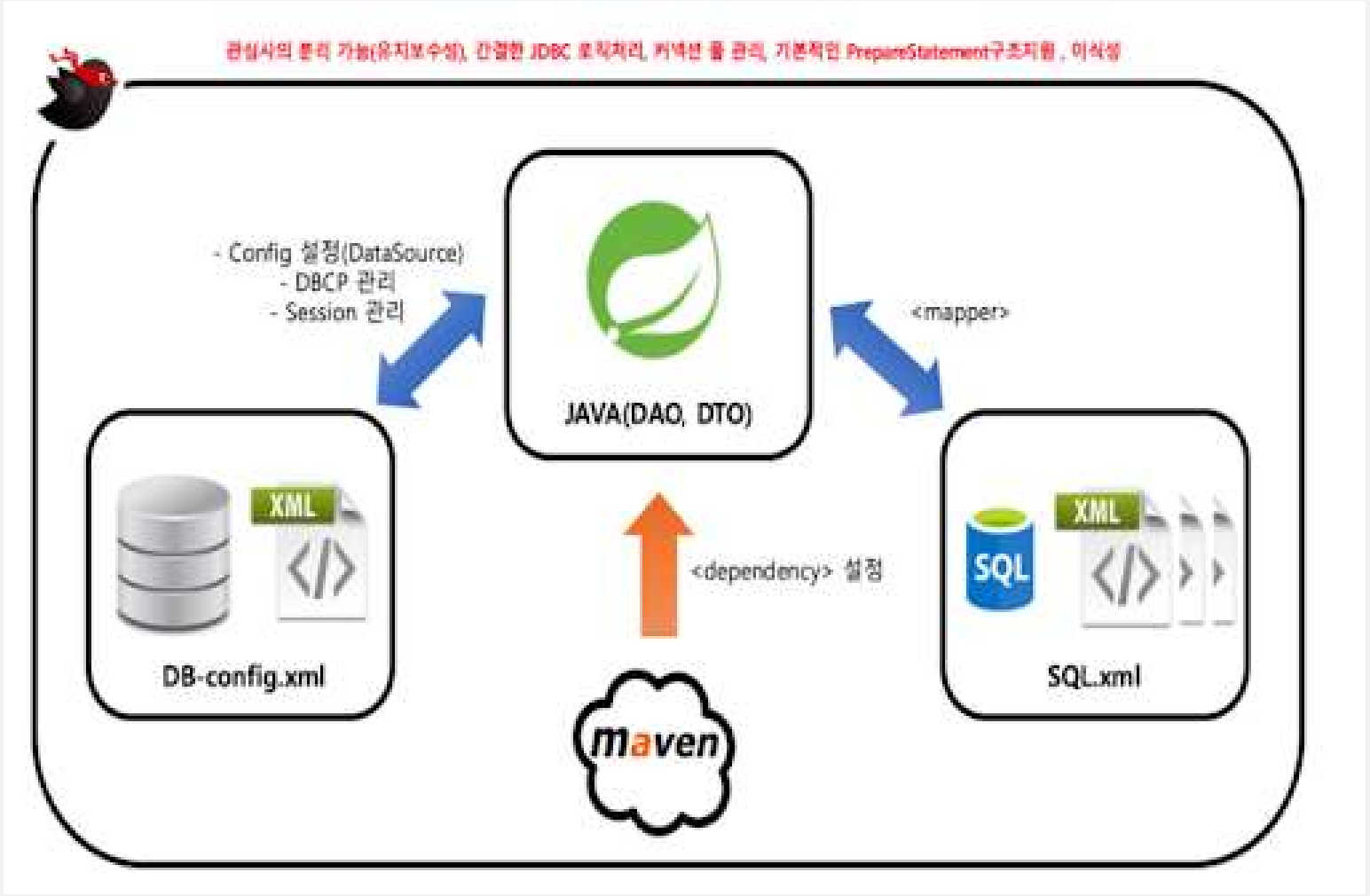
쉽게 말해 JDBC를 사용하기 편리하게 소스코드(JAVA)와 SQL(XML)을 분리하여 개발 할 수 있게 해주는 프레임워크이다.

ORM이란?

ORM이란 객체(**O**bject)와 관계(**R**elation)을 연결(**M**apping)해 주는 개념이다.
객체와 테이블 시스템(RDBMS)을 변형 및 연결해주는 작업이다.

가장많이 사용하는 ORM은 **Mybatis**와 **Hibernate(JPA)**가 대표적이다

Mybatis 구조


관심사의 분리 가능(유지보수성), 간결한 JDBC 로직처리, 커넥션 풀 관리, 기본적인 PrepareStatement구조지원 , 이식성
- Config 설정(DataSource)
- DBCP 관리
- Session 관리
JAVA(DAO, DTO)
<mapper>
XML
DB-config.xml
<dependency> 설정
maven
SQL
XML
SQL.xml

Mybatis-config

```xml
<?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="${driver}"/>
        <property name="url" value="${url}"/>
        <property name="username" value="${username}"/>
        <property name="password" value="${password}"/>
      </dataSource>
    </environment>
  </environments>
  <mappers>
    <mapper resource="org/mybatis/example/BlogMapper.xml"/>
  </mappers>
</configuration>
```

```xml
<typeAliases>
  <typeAlias alias="Author" type="domain.blog.Author"/>
  <typeAlias alias="Blog" type="domain.blog.Blog"/>
  <typeAlias alias="Comment" type="domain.blog.Comment"/>
  <typeAlias alias="Post" type="domain.blog.Post"/>
  <typeAlias alias="Section" type="domain.blog.Section"/>
  <typeAlias alias="Tag" type="domain.blog.Tag"/>
</typeAliases>
```

```xml
<dataSource type="JNDI"> <!-- JNDI로 한다. -->
  <!-- data_source으로 JNDI부터 조회할 수 있는 이름을 지정한다. -->
  <property name="data_source" value="java:app/sampleDS"/>
</dataSource>
```

Mybatis-connector

```java
public class MybatisConnector {

    private SqlSessionFactory factory = null;
    private static MybatisConnector connector;

    private MybatisConnector() {
        try {
            Reader reader;
            reader = Resources.getResourceAsReader("confit/mybatis-config.xml");
            factory = new SqlSessionFactoryBuilder().build(reader);
        } catch (IOException e) {
            e.printStackTrace();
        }

    }

    public static MybatisConnector getInstance() {
        if(connector==null) {
            connector = new MybatisConnector();
        }
        return connector;
    }

    public SqlSessionFactory getSqlSessionFactory() {
        return factory;
    }

}
```

Mybatis-mapper

```xml
<?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="org.mybatis.example.BlogMapper">
  <select id="selectBlog" resultType="Blog">
    select * from Blog where id = #{id}
  </select>
</mapper>
```

Mybatis-mapper

```
<select id="selectPerson" parameterType="int" resultType="hashmap">
  SELECT * FROM PERSON WHERE ID = #{id}
</select>
```

```
<select id="selectUsers" resultType="User">
  select id, username, password
  from users
  where id = #{id}
</select>
```

객체 사용가능

```
<insert id="insertUser" parameterType="User">
  insert into users (id, username, password)
  values (#{id}, #{username}, #{password})
</insert>
```

Mybatis-mapper

```
<insert id="insertAuthor">
  insert into Author (id,username,password,email,bio)
  values (#{id},#{username},#{password},#{email},#{bio})
</insert>

<update id="updateAuthor">
  update Author set
    username = #{username},
    password = #{password},
    email = #{email},
    bio = #{bio}
  where id = #{id}
</update>

<delete id="deleteAuthor">
  delete from Author where id = #{id}
</delete>
```

Mybatis-DAO

```java
public class UserDAO {
    private static UserDAO dao = null;
    private SqlSessionFactory factory=null;

    private UserDAO() {
        factory = MybatisConnector.getInstance().getSqlSessionFactory();
    }

    public static UserDAO getInstance() {
        if(dao == null) {
            dao = new UserDAO();
        }
        return dao;
    }

    public List<UserVO> selectList(){
        List<UserVO> list = null;
        SqlSession sqlSession = factory.openSession();
        list=sqlSession.selectList("user.selectUsers");
        sqlSession.close();
        return list;
    }
}
```

openSession(false)

오토커밋 해제

sqlSession.commit();
sqlSession.rollback();

Mybatis-mapper :
reseultMap

```
<resultMap id="userResultMap" type="User">
  <id property="id" column="user_id" />
  <result property="username" column="username"/>
  <result property="password" column="password"/>
</resultMap>
```

```
<select id="selectUsers" resultMap="userResultMap">
```

Mybatis-mapper : 동적 SQL

```
<select id="findActiveBlogLike"
     resultType="Blog">
  SELECT * FROM BLOG WHERE state = ‘ACTIVE’
  <if test="title != null">
    AND title like #{title}
  </if>
  <if test="author != null and author.name != null">
    AND author_name like #{author.name}
  </if>
</select>
```

```
<select id="findActiveBlogLike"
     resultType="Blog">
  SELECT * FROM BLOG WHERE state = ‘ACTIVE’
  <choose>
    <when test="title != null">
      AND title like #{title}
    </when>
    <when test="author != null and author.name != null">
      AND author_name like #{author.name}
    </when>
    <otherwise>
      AND featured = 1
    </otherwise>
  </choose>
</select>
```