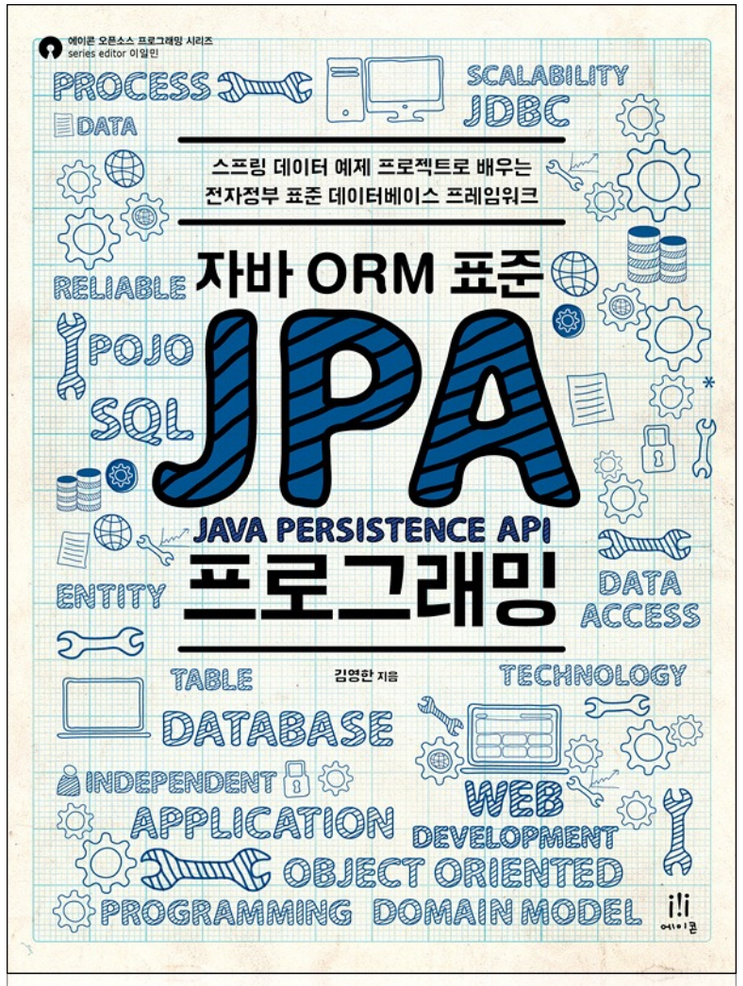
JPA 시작

Hello JPA 실습

김영한

SI, J2EE 강사, DAUM, SK 플래닛
우아한형제들

저서: 자바 ORM 표준 **JPA** 프로그래밍



목차

- Hello JPA - 프로젝트 생성
- Hello JPA - 애플리케이션 개발

Hello JPA - 프로젝트 생성

H2 데이터베이스 설치와 실행

- http://www.h2database.com/
- 최고의 실습용 DB
- 가볍다.(1.5M)
- 웹용 쿼리툴 제공
- MySQL, Oracle 데이터베이스 시뮬레이션 기능
- 시퀀스, AUTO INCREMENT 기능 지원

메이븐 소개

- https://maven.apache.org/
- 자바 라이브러리, 빌드 관리
- 라이브러리 자동 다운로드 및 의존성 관리
- 최근에는 그래들(Gradle)이 점점 유명

프로젝트 생성

- **자바 8 이상(8 권장)**
- **메이븐 설정**
  - **groupId**: jpa-basic
  - **artifactId**: ex1-hello-jpa
  - **version**: 1.0.0

라이브러리 추가 - pom.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>jpa-basic</groupId>
    <artifactId>ex1-hello-jpa</artifactId>
    <version>1.0.0</version>

    <dependencies>
        <!-- JPA 하이버네이트 -->
        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-entitymanager</artifactId>
            <version>5.3.10.Final</version>
        </dependency>

        <!-- H2 데이터베이스 -->
        <dependency>
            <groupId>com.h2database</groupId>
            <artifactId>h2</artifactId>
            <version>1.4.199</version>
        </dependency>
    </dependencies>

</project>
```

JPA 설정하기 - persistence.xml

- JPA 설정 파일
- /META-INF/persistence.xml 위치
- persistence-unit name으로 이름 지정
- javax.persistence로 시작: JPA 표준 속성
- hibernate로 시작: 하이버네이트 전용 속성

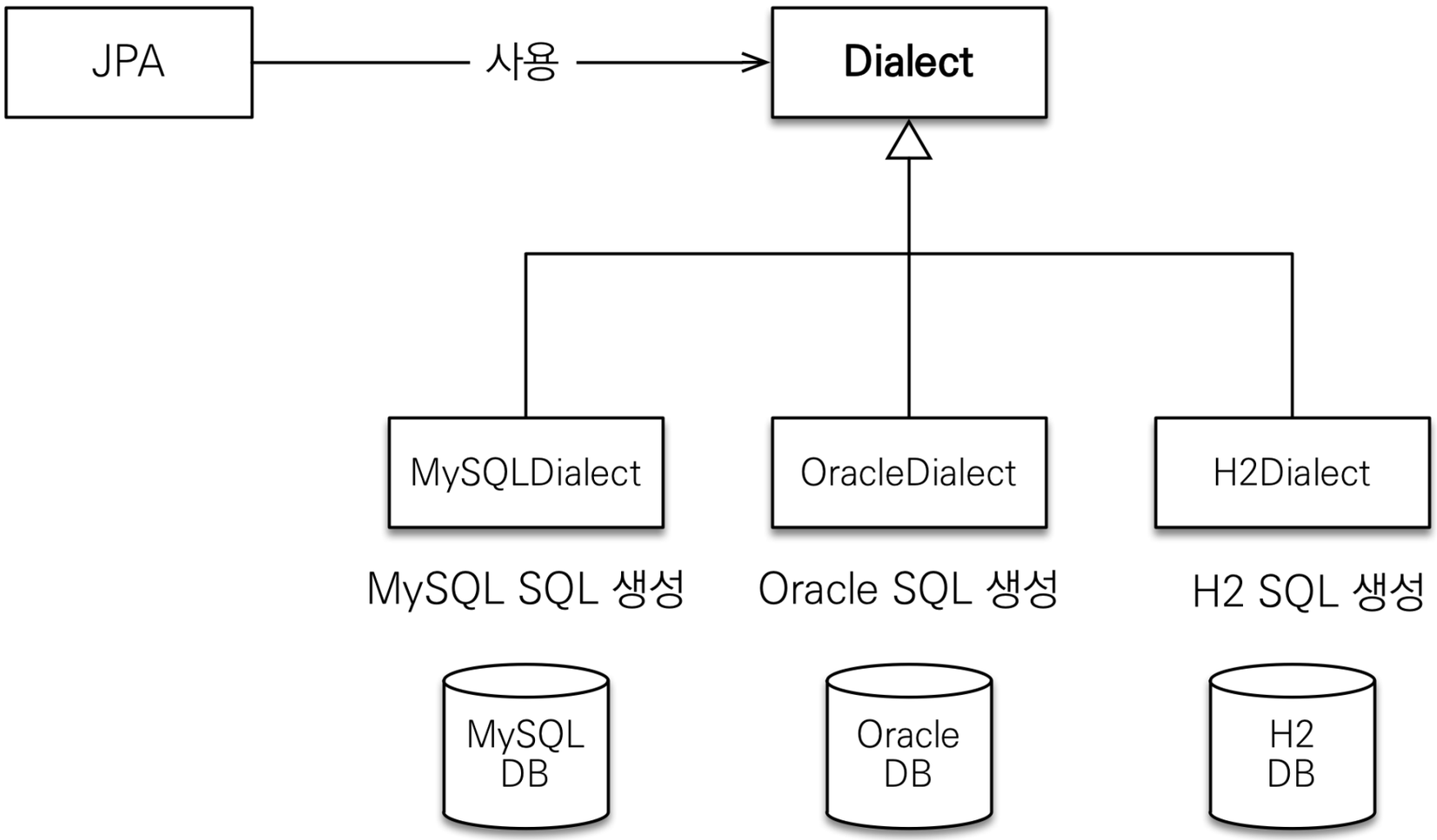
persistence.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.2"
             xmlns="http://xmlns.jcp.org/xml/ns/persistence" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence http://xmlns.jcp.org/xml/ns/persistence/persistence_2_2.xsd">

    <persistence-unit name="hello">
        <properties>
            <!-- 필수 속성 -->
            <property name="javax.persistence.jdbc.driver" value="org.h2.Driver"/>
            <property name="javax.persistence.jdbc.user" value="sa"/>
            <property name="javax.persistence.jdbc.password" value=""/>
            <property name="javax.persistence.jdbc.url" value="jdbc:h2:tcp://localhost/~/test"/>
            <property name="hibernate.dialect" value="org.hibernate.dialect.H2Dialect"/>

            <!-- 옵션 -->
            <property name="hibernate.show_sql" value="true"/>
            <property name="hibernate.format_sql" value="true"/>
            <property name="hibernate.use_sql_comments" value="true"/>
            <!--<property name="hibernate.hbm2ddl.auto" value="create" />-->
        </properties>
    </persistence-unit>
</persistence>
```

데이터베이스 방언

- JPA는 특정 데이터베이스에 종속 X
- 각각의 데이터베이스가 제공하는 SQL 문법과 함수는 조금씩 다름
  - 가변 문자: MySQL은 VARCHAR, Oracle은 VARCHAR2
  - 문자열을 자르는 함수: SQL 표준은 SUBSTRING(), Oracle은 SUBSTR()
  - 페이징: MySQL은 LIMIT , Oracle은 ROWNUM
- 방언: SQL 표준을 지키지 않는 특정 데이터베이스만의 고유한 기능

데이터베이스 방언

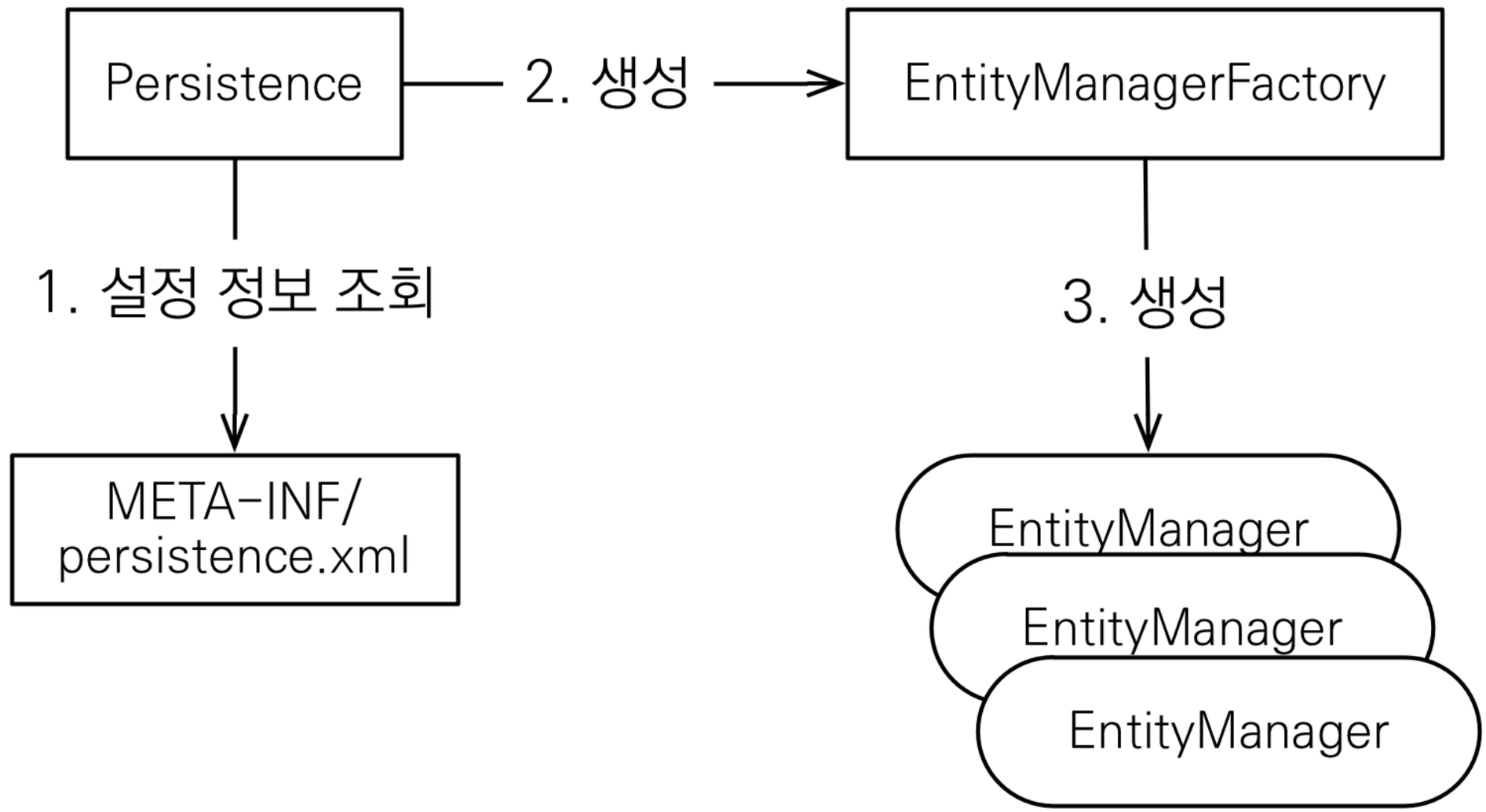

JPA
사용
Dialect
MySQLDialect
OracleDialect
H2Dialect
MySQL SQL 생성
Oracle SQL 생성
H2 SQL 생성
MySQL DB
Oracle DB
H2 DB

데이터베이스 방언

- **hibernate.dialect** 속성에 지정
  - H2 : org.hibernate.dialect.H2Dialect
  - Oracle 10g : org.hibernate.dialect.Oracle10gDialect
  - MySQL : org.hibernate.dialect.MySQL5InnoDBDialect
- 하이버네이트는 40가지 이상의 데이터베이스 방언 지원

Hello JPA - 애플리케이션 개발

JPA 구동 방식


Persistence
2. 생성
EntityManagerFactory
1. 설정 정보 조회
3. 생성
META-INF/
persistence.xml
EntityManager
EntityManager
EntityManager

실습 - JPA 동작 확인

- JpaMain 클래스 생성
- JPA 동작 확인

객체와 테이블을 생성하고 매핑하기

- **@Entity:** JPA가 관리할 객체
- **@Id**: 데이터베이스 PK와 매핑

```java
package hellojpa;

import javax.persistence.Entity;
import javax.persistence.Id;

@Entity
public class Member {

    @Id
    private Long id;
    private String name;

    //Getter, Setter …
}
```

```sql
create table Member (

    id bigint not null,
    name varchar(255),
    primary key (id)
);
```

실습 - 회원 저장

- 회원 등록
- 회원 수정
- 회원 삭제
- 회원 단 건 조회

주의

- **엔티티 매니저 팩토리**는 하나만 생성해서 애플리케이션 전체에서 공유
- **엔티티 매니저**는 쓰레드간에 공유X (사용하고 버려야 한다).
- **JPA의 모든 데이터 변경은 트랜잭션 안에서 실행**

JPQL 소개

- 가장 단순한 조회 방법
  - EntityManager.find()
  - 객체 그래프 탐색(a.getB().getC())
- 나이가 18살 이상인 회원을 모두 검색하고 싶다면?

실습 - JPQL 소개

- JPQL로 전체 회원 검색
- JPQL로 ID가 2 이상인 회원만 검색
- JPQL로 이름이 같은 회원만 검색
- JPQL에 대해 자세한 내용은 객체지향 쿼리에서 학습

JPQL

- JPA를 사용하면 엔티티 객체를 중심으로 개발
- 문제는 검색 쿼리
- 검색을 할 때도 테이블이 아닌 엔티티 객체를 대상으로 검색
- 모든 DB 데이터를 객체로 변환해서 검색하는 것은 불가능
- 애플리케이션이 필요한 데이터만 DB에서 불러오려면 결국 검색 조건이 포함된 SQL이 필요

JPQL

- JPA는 SQL을 추상화한 JPQL이라는 객체 지향 쿼리 언어 제공
- SQL과 문법 유사, SELECT, FROM, WHERE, GROUP BY, HAVING, JOIN 지원
- **JPQL은 엔티티 객체**를 대상으로 쿼리
- **SQL은 데이터베이스 테이블**을 대상으로 쿼리

JPA와 SQL의 차이점

객체를 대상으로 한 덕분에 JPA는 벤더한 관계없이 방언(SQL, 오라클)을 바꾸더라도 잘 작동한다.

JPQL

- 테이블이 아닌 **객체를 대상으로 검색하는 객체 지향 쿼리**
- SQL을 추상화해서 특정 데이터베이스 SQL에 의존X → SQL이 오라클, h2 다 번역O
- JPQL을 한마디로 정의하면 객체 지향 SQL
- **JPQL은 뒤에서 아주 자세히 다룸**