Pemrograman Framework Java

Pertemuan 4 – Akses Database

Akses Database Spring

- Spring memiliki fitur untuk mengakses database
- Pada dasarnya Java juga memiliki kemampuan untuk melakukan koneksi ke database melalui library
- Namun Spring sudah menyediakan apa yang dibutuhkan

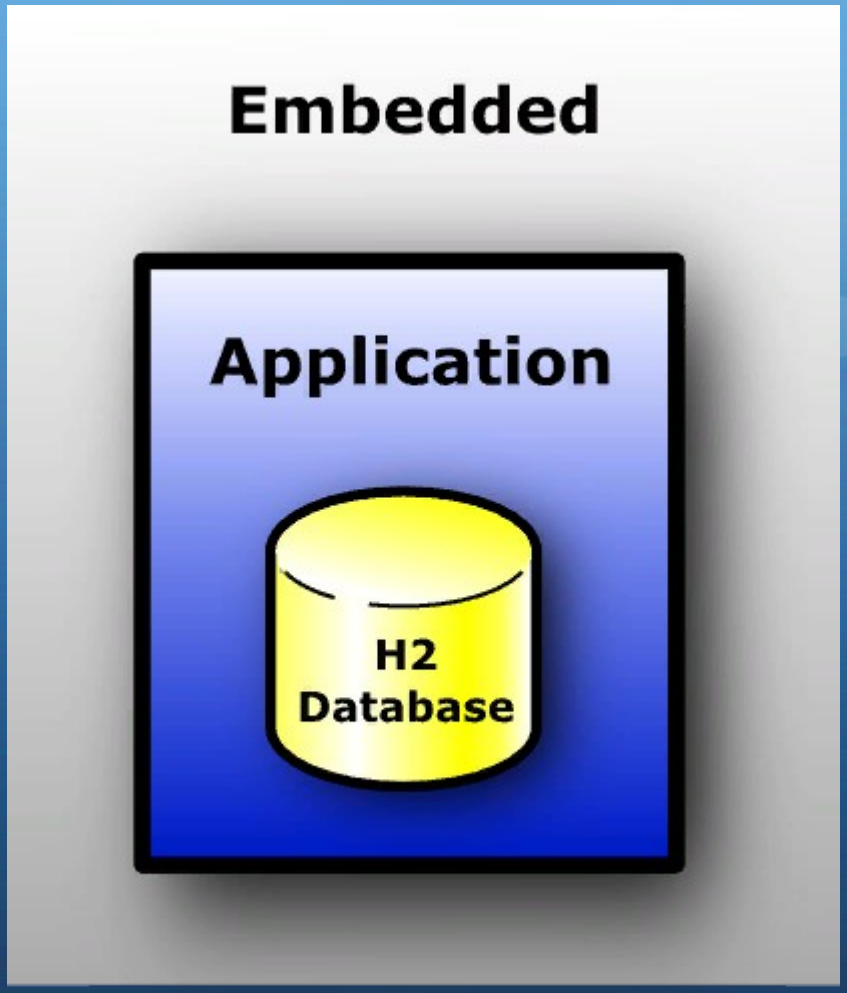
Dukungan Database

- Spring mendukung dua jenis database
  - Database Internal (Embed)
  - Database External
- Hal ini untuk dibuat untuk memudahkan programmer membuat aplikasi

Database Internal

- Database internal/embed adalah database yang memiliki integrasi kuat dengan aplikasi software
- Database ini hanya berjalan jika aplikasi tersebut berjalan
- Spring sudah mendukung beberapa db secara native

Ilustrasi



Mengapa DB Internal?

- Database internal sangat berguna ketika fase pengembangan projek karena ringan.
- Keuntungan yang ada di dalamnya berupa kemudahan konfigurasi, waktu startuo cepat, kemudahan pengujian, dan kemudahan untuk evolusi SQL selama pengembangan.

Pembuatan DB

- DB di dalam Spring bisa dibuat dengan dua cara
  - Dengan menggunakan XML
  - Dengan cara pemrograman

Dengan XML

- Ketika ingin mendefinisikan JDBC / koneksi DB sebagai beans, maka tag spring-jdbc harus digunakan

```
<jdbc:embedded-database id="dataSource">
   <jdbc:script location="classpath:schema.sql"/>
   <jdbc:script location="classpath:test-data.sql"/>
</jdbc:embedded-database>
```

Dengan Pemrograman

- Pendefinisian koneksi JDBC menggunakan Java class. Berikut adalah contoh kode pendefinisan

```
EmbeddedDatabaseBuilder builder = new EmbeddedDatabaseBuilder();
    EmbeddedDatabase db = builder.setType(H2).addScript("my-schema.sql").addScript("my-test-data.sql").build();
    // do stuff against the db (EmbeddedDatabase extends javax.sql.DataSource)
    db.shutdown()
```

Schema & Test-Data

- Pendefinisian baik dengan XML maupun Pemrograman, keduanya melakukan panggilan ke schema, dan test-data
- Kedua file ini berisi perintah dalam SQL untuk melakukan
  - Pembuatan skema database
  - Data buatan ke dalam database tersebut

Contoh Schema.SQL

- CREATE TABLE cars(id INT PRIMARY KEY AUTO_INCREMENT, name VARCHAR(150), price INT);

- Script ini berguna untuk membuat table mobil dengan parameter yang telah ditentukan

Contoh Test-data.SQL

- INSERT INTO cars(name, price) VALUES('Audi', 52642);
- INSERT INTO cars(name, price) VALUES('Mercedes', 57127);
- INSERT INTO cars(name, price) VALUES('Skoda', 9000);
- INSERT INTO cars(name, price) VALUES('Volvo', 29000);
- INSERT INTO cars(name, price) VALUES('Bentley', 350000);
- INSERT INTO cars(name, price) VALUES('Citroen', 21000);
- INSERT INTO cars(name, price) VALUES('Hummer', 41400);
- INSERT INTO cars(name, price) VALUES('Volkswagen', 21600);

Keuntungan

- Tidak memerlukan koneksi setiap waktu menggunakan database
- Data tersimpan secara internal di dalam aplikasi
- Portabilitas
- Dan Mobilitas

Kerugian

- Kerawanan data bersama dengan aplikasi
- Untuk mengakses database, aplikasi harus dijalankan terlebih dahulu

Daftar Embed DB

- **HSQLDB from HSQLDB.ORG,**
- **H2SQLite**
- **Derby**
- SQL Server Compact from Microsoft Corporation

Database dengan **BOLD** didukung oleh Spring Data

DB Eksternal

- Database ini terletak di luar dari aplikasi, sehingga akses nya memerlukan koneksi ke DB server
- DB Eksternal digunakan untuk berbagai macam keperluan, sehingga aksesnya tidak hanya dari satu aplikasi saja

Kelebihan & Kekurangan

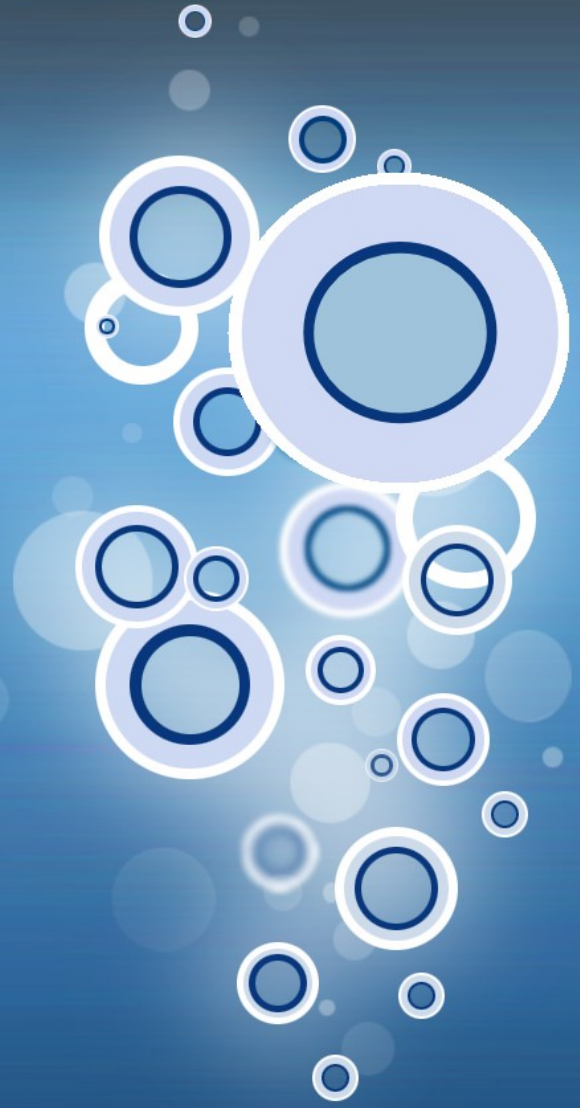
- Kelebihan dan Kekurangan dari DB Eksternal adalah kebalikan dari DB Internal
- Sehingga perlu dipikirkan ketika akan menggunakan DB Internal maupun DB Eksternal

Dukungan DB

- Pada dasarnya Spring akan mendukung semua DB Eksternal yang ada
- Namun tentunya diperlukan konfigurasi ekstra untuk mendukung

Lanjutan

- MySQL/MariaDB
- PostgreSQL
- Oracle
- DB2
- ETC



Pengaksesan

- Pengaksesan DB baik internal maupun eksternal memerlukan JDBC Template dari Spring agar mempermudah penggunaan
- JDBC Template sudah disediakan secara default oleh Spring, sehingga kita cukup melakukan import saja

Mengapa JDBC Template?

- JDBC Template menyediakan sebuah arti untuk pengembang untuk melakukan operasi SQL terhadap database tanpa kerumitan ketika bekerja dengan JDBC biasa.
- Karena tugas dasar Template adalah menyediakan cetakan dasar untuk mempermudah penggunaan

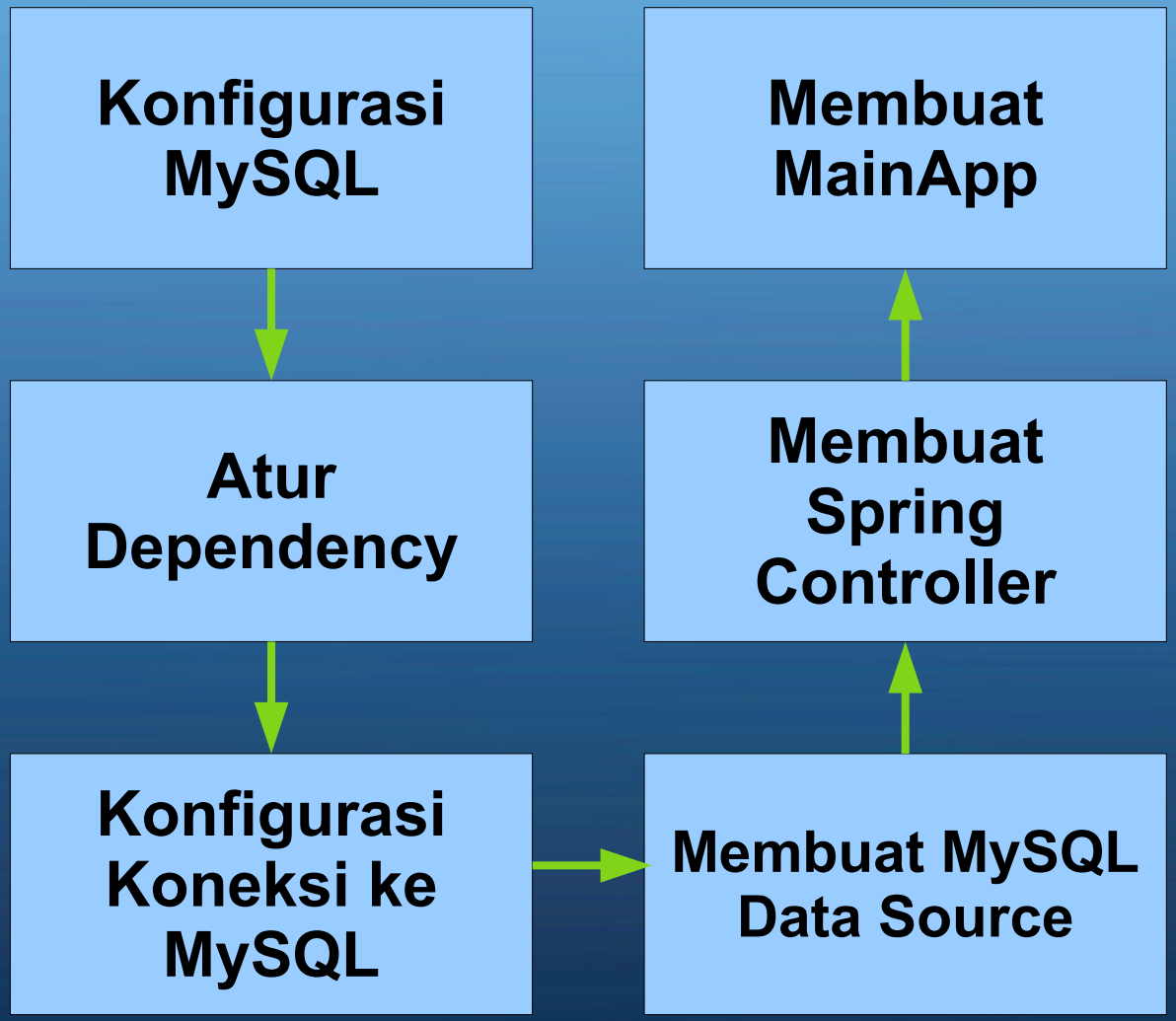
Contoh JDBC Template

```
private JdbcTemplate jdbc;

@Override
public Ingredient findOne(String id) {
  return jdbc.queryForObject(
      "select id, name, type from Ingredient where id=?",
      this::mapRowToIngredient, id);
}

private Ingredient mapRowToIngredient(ResultSet rs, int rowNum)
    throws SQLException {
  return new Ingredient(
      rs.getString("id"),
      rs.getString("name"),
      Ingredient.Type.valueOf(rs.getString("type")));
}
```

Menggunakan JDBC

- Hal yang paling mendasar ketika ingin menggunakan JDBC adalah mendeklarasikan kebutuhan (dependency) akan database dan JDBC melalui pom.xml
- Semua library yang dibutuhkan akan di download oleh Maven

Ilustrasi Proses



Konfigurasi MySQL

- Konfigurasi yang dimaksud adalah pemberian username dan password pada database
- Koneksi yang digunakan localhost/remote server
- Dan DB yang akan digunakan

Dependency MySQL

- Cukup dengan menambahkan

```
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
</dependency>
```

Konfigurasi Koneksi

- server.port=8086
- server.contextPath=/myapp
- spring.main.banner-mode=off
- logging.level.org.springframework=ERROR
- spring.datasource.url=jdbc:mysql://
  localhost:3306/testdb?useSSL=false
- spring.datasource.username=user12
- spring.datasource.password=s$cret

DB Data Source

- Di bagian ini kita membuat jenis data yang ingin dimasukkan ke DB
- Berbentuk class yang isi dari Entity Object mirip dengan Tabel di DB
- Dan juga Repository class harus dibuat

Spring Controller

- Controller berguna untuk melakukan akses baca tulis ke/dari DB
- Kita bisa melakukan manipulasi data di DB menggunakan Controller class ini, dan melakukan penampilan dalam bentuk tertentu

Membuat MainApp

- MainApp adalah syarat utama untuk aplikasi Java untuk berjalan, oleh karena itu tidak boleh dilupakan untuk dibuat.
- MainApp akan menjalankan aplikasi Spring ketika dijalankan

Lanjutan

- Tahap-tahap tadi bisa dipraktikan di DB lainnya
- Seperti:
  - H2
  - Derby
  - HSQL
  - PostgreSQL
  - dll

Kuis Pengganti Kelas

- Jelaskan Apa Itu Framework, dan Keuntungan Menggunakan Framework!
- Jelaskan Apa Tugas dan Fungsi pom.xml dalam projek Spring!
- Jelaskan Urutan Build Life Cycle dari Maven!
- Jelaskan Keuntungan Menggunakan Spring Boot!

Kuis Pengganti Kelas

- Jelaskan Apa Itu Dependency!
- Jelaskan Apa itu Tight dan Loose Coupling
-