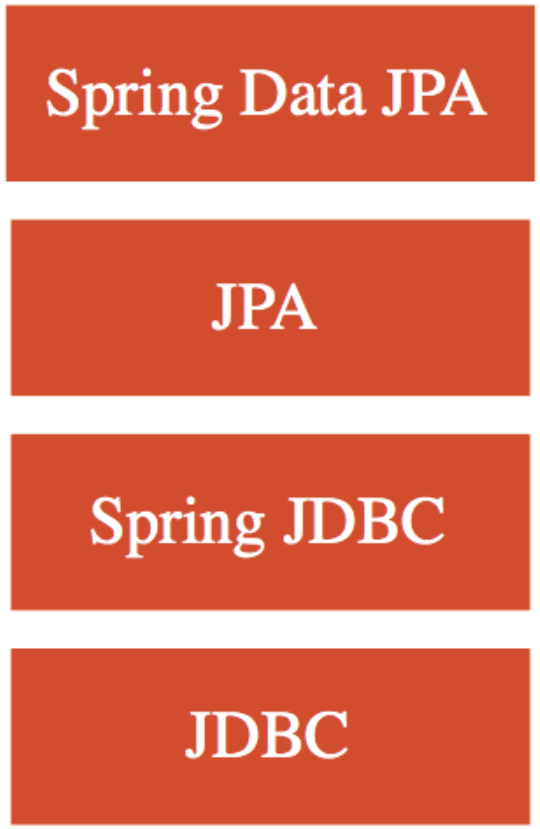
In28 Minutes

JPA and Hibernate in 10 Steps

Getting Started with JPA and Hibernate

In 28 Minutes

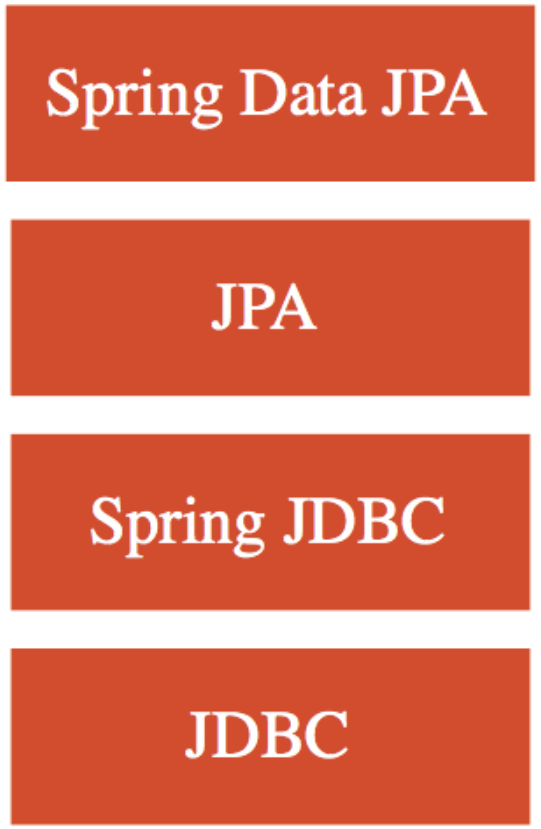
- Build a Simple JPA App using **Modern Spring Boot** Approach
- Get **Hands-on** with JPA, Hibernate and Spring Boot
  - World before JPA - JDBC, Spring JDBC
  - Why JPA? Why Hibernate? (JPA vs Hibernate)
  - Why Spring Boot and Spring Boot Data JPA?
  - **JPA** Terminology: Entity and Mapping



Learning JPA and Hibernate - Approach

In 28 Minutes

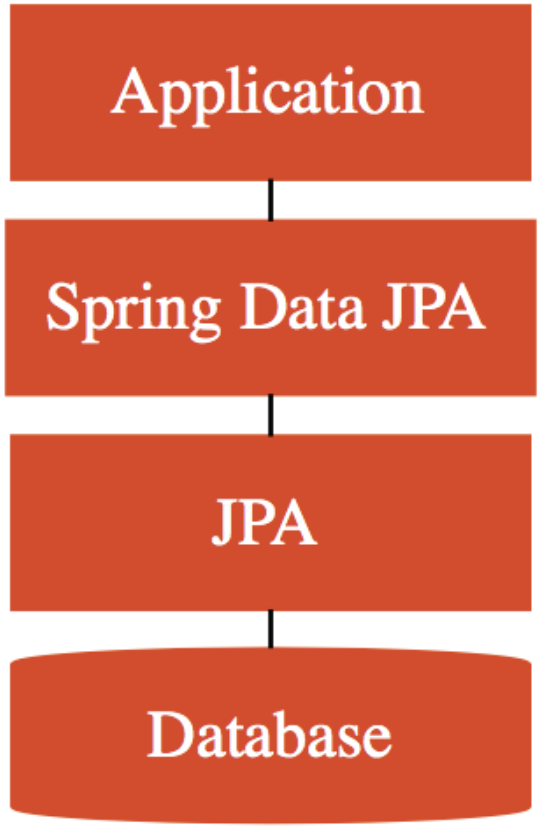
- 01: Create a **Spring Boot Project** with H2
- 02: Create **COURSE table**
- 03: Use **Spring JDBC** to play with COURSE table
- 04: Use **JPA and Hibernate** to play with COURSE table
- 05: Use **Spring Data JPA** to play with COURSE table



Spring Boot Auto Configuration Magic

In28 Minutes

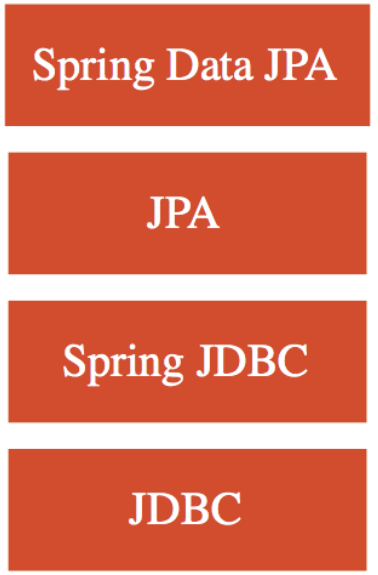
- We added Data JPA and H2 dependencies:
  - Spring Boot Auto Configuration does some magic:
    - Initialize JPA and Spring Data JPA frameworks
    - Launch an in memory database (H2)
    - Setup connection from App to in-memory database
    - Launch a few scripts at startup (example: `data.sql`, `schema.sql`)
- **Remember** - H2 is in memory database
  - Does NOT persist data
  - Great for learning
  - BUT NOT so great for production



JDBC to Spring JDBC to JPA to Spring Data JPA

In 28 Minutes

- **JDBC**
  - Write a lot of SQL queries! (*delete from todo where id=?*)
  - And write a lot of Java code
- **Spring JDBC**
  - Write a lot of SQL queries (*delete from todo where id=?*)
  - BUT lesser Java code
- **JPA**
  - Do NOT worry about queries
  - Just Map Entities to Tables!
- **Spring Data JPA**
  - Let's make JPA even more simple!
  - I will take care of everything!



JDBC to Spring JDBC

In28 Minutes

JDBC example

```
public void deleteTodo(int id) {
    PreparedStatement st = null;
    try {
        st = db.conn.prepareStatement("delete from todo where id=?");
        st.setInt(1, id);
        st.execute();
    } catch (SQLException e) {
        logger.fatal("Query Failed : ", e);
    } finally {
        if (st != null) {
            try {st.close();}
            catch (SQLException e) {}
        }
    }
}
```

Spring JDBC example

```
public void deleteTodo(int id) {
    jdbcTemplate.update("delete from todo where id=?", id);
}
```

JPA Example

```
@Repository
public class PersonJpaRepository {

  @PersistenceContext
  EntityManager entityManager;

  public Person findById(int id) {
    return entityManager.find(Person.class, id);
  }

  public Person update(Person person) {
    return entityManager.merge(person);
  }

  public Person insert(Person person) {
    return entityManager.merge(person);
  }

  public void deleteById(int id) {........
```

Spring Data JPA Example

```
public interface TodoRepository extends JpaRepository<Todo, Integer>{
```

Hibernate vs JPA

- **JPA** defines the specification. It is an API.
  - How do you define entities?
  - How do you map attributes?
  - Who manages the entities?
- Hibernate is one of the popular implementations of JPA
- Using Hibernate directly would result in a lock in to Hibernate
  - There are other JPA implementations (Toplink, for example)