SPRING BOOT

[CREATING A SIMPLE SPRING BOOT PROJECT:](#_jnuszi4q7vj)

[SPRING BOOT WITH REST:](#_hi25ijjf4n19)

[ANNOTATIONS:](#_p3yhmmrhbn5y)

[JPA :](#_s3sgkeqmh64p)

[JPA REPOSITORY:](#_vrmqf1pemhr6)

[METHODS IN JPA REPOSITORY:](#_eogczq2j7jep)

[DEFINING CUSTOM METHODS IN REPOSITORY:](#_nc4dnwhtlb5x)

[EXERCISE:](#_7tpvshziel7e)

[STEPS:](#_866lwq9gwgm0)

[SOURCE CODE:](#_j405ewclacnc)

Spring Boot is a Spring framework module which provides RAD (Rapid Application Development) feature to the Spring framework. It is highly dependent on the **starter templates** feature which is very powerful and works flawlessly.

STARTER TEMPLATES:

Spring Boot starters are templates that contain a collection of all the relevant transitive dependencies that are needed to start a particular functionality.

With String boot, to create MVC application all you need to import is spring-boot-starter-web dependency.

pom.xml

<!-- Parent pom is mandatory to control versions of child dependencies -->  
<parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>2.0.4.RELEASE</version>  
 <relativePath />  
</parent>  
   
<!-- Spring web brings all required dependencies to build web application. -->  
<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
</dependency>

# CREATING A SIMPLE SPRING BOOT PROJECT:

Spring boot project can be initialized in 3 ways

* Using Spring Tool Suite (STS)
  + [Documentation](https://spring.io/guides/gs/sts/)
  + [Tutorial video](https://www.youtube.com/watch?v=3_AnXmsh90g&amp=&index=3&amp=&list=PLsyeobzWxl7oA8QOlMtQsRT_I7Rx2hoX4)
* Using Spring Initializr
  + [spring initializr](https://start.spring.io/)
* Using Maven
  + [Documentation](https://spring.io/guides/gs/spring-boot/)
  + [Tutorial video](https://www.youtube.com/watch?v=bDtZvYAT5Sc)

## SPRING BOOT WITH REST:

In our application, we have 3 layer architecture

1. Controller

2. Delegate

3. DAO

### ANNOTATIONS:

@Autowired

This annotation is used to carry out Dependency Injection. It is used to inject the

singleton object of the required class

*@EnableAutoConfiguration*

This annotation enables auto-configuration. It means that **Spring Boot looks for auto-configuration beans** on its classpath and automatically applies them.

@RestController

For a RESTful web service, the controller is annotated with

@RestController(which in turn contains @Controller and @ResponseBody),

which tells Spring that the response should be bound to the web response body.

@Service

Delegate is annotated with @Service which tells spring to create a singleton

object and use the same for dependency injection.

@Service denotes that this class is used for business specific logic.

@Repository

DAO is annotated with @Repository which tells spring to create a singleton object and use the same for dependency injection.

@Repository denotes that the class is used as a data access layer.

## 

## JPA :

* Java Persistence API - a specification to implement ORM ( Object Relation Mapping ) - to define entities, map attributes, map relationships between entities and manage entities.
* It uses entities - basic persistence units ( to store data ) and every entity class should have an @Entity marker and an identifier field, indicated by @Id, that is mapped to the primary key column in the database.
* Each attribute in the entity is annotated with @Column to indicate it is a database column.

Where to use JPA?

To reduce the burden of writing codes for relational object management, a programmer follows the ‘JPA Provider’ framework, which allows easy interaction with database instance. Here the required framework is taken over by JPA.

### 

### JPA REPOSITORY:

● A Repository is an interface which holds in-built implementation for storage,

retrieval, search, update and delete operation on objects.

● CrudRepository provides basic implementation for Create-Read-Update-Delete

operations on an entity.

public interface EmployeeRepository extends JpaRepository<User,Long> { }

### 

### METHODS IN JPA REPOSITORY:

There are many inbuilt methods for database operations.

Some of them are:

save() - to create a new entry in a table

findAll() - to retreive all entries in a table

deleteById() - to delete an entry by Id in a table

findById() - to identify an entry by Id.

<https://docs.spring.io/spring-data/jpa/docs/current/api/org/springframework/data/jpa/repository/JpaRepository.html>

### DEFINING CUSTOM METHODS IN REPOSITORY:

@Repository

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

@Query("SELECT CASE WHEN COUNT(c) > 0 THEN true ELSE false END FROM Employee c WHERE c.email = :email")

boolean existsByEmail(@Param("email") String email);

}

The method existsByEmail will return a boolean value indicating if the email exists in the employee table.Similarly many custom methods can be implemented using @Query Annotation.

## 

## EXERCISE:

Implement CRUD operation on the given employee table using JPA. The schema for the table is

given below.

create table employee(id int auto\_increment primary key, firstname varchar(40), lastname

varchar(40), email varchar(50) unique, phone\_number bigint, salary bigint);

### STEPS:

* Create separate packages for beans,controller,dao,delegate,repository.
* Repository package must contain an interface for employee that extends JPARepository which inturn extends CRUDRepository.
* Controller consists of different methods for performing CRUD operations.
* The controller makes a call to delegate and delegate to DAO.
* In DAO instead of writing queries for each database operation the instance of Employee repository is used to access all the jpa methods.
* In src/main/resources, create a file named “application.properties” and define port-number, and database connection essentials.

server.port = 8081

spring.datasource.url= jdbc:mysql://localhost:3306/SpringBoot

spring.datasource.username=root

spring.datasource.password=root@123

* Add Jpa dependencies in pom.xml.
* Create a java file in the src package that will start the spring boot application.

### SOURCE CODE:

<https://github.com/sruthiviswanathan/Zterns-19-Sruthi/tree/master/SpringBootCrud>