

Spring Core Assignments

- 1) Create an Address class with the following attributes:- street, city, state, zip, country
Create an Customer class with the following attributes:- customerId, customerName, customerContact, customerAddress.
Inject the Address bean into Customer bean using setter injection
Create a Test class with main() method, get Customer bean from ApplicationContext object and print details of Customer.
Also write the JUnit Test cases for above program.
 - Modify the above application and inject the bean using constructor injection
 - Use XML based Configuraion.

2) Example of Injecting collections (List, Set and Map)

Create a class Question with following attributes: questionId, question, answers.

There are 3 cases for above program.

- a. Write a program where answers is of type List<String> or String []
 - b. Write a program where answers is of type Set<String>
 - c. Write a program where answers is of type Map<Integer, String>
In case of Map, Integer value represents answer's sequence number.
 - d. Create a Test class with main() method, get Question bean from ApplicationContext object and print question and its answers.
 - e. Also write the JUnit Test cases for above program.
- Use XML based configuration.

3) Example on autowiring

Design and Develop a Banking Application as follows:

- a. Create a BankAccount class with following attributes: accountId, accountHolderName, accountType, accountBalance
- b. Create an interface BankAccountRepository with following methods:
public double getBalance(long accountId)
public double updateBalance(long accountId, double newBalance):
Note: Above method returns updated balance.

- c. Create a class `BankAccountRepositoryImpl` that implements `BankAccountRepository` interface.
You can use database or any collection object as persistence store.
- d. Create an interface `BankAccountService` with following methods:
`public double withdraw(long accountId, double balance)`
`public double deposit(long accountId, double balance)`
`public double getBalance(long accountId)`
`public boolean fundTransfer(long fromAccount, long toAccount, double amount)`
- e. Create a class `BankAccountServiceImpl` that implements `BankAccountService` interface.
- f. Create a class `BankAccountController` with following operations:
`public double withdraw(long accountId, double balance)`
`public double deposit(long accountId, double balance)`
`public double getBalance(long accountId)`
`public boolean fundTransfer(long fromAccount, long toAccount, double amount)`

- g. Create a Test class with main() method, get BankAccountController bean object from ApplicationContext and perform all the operations.
 - h. Also write the JUnit Test cases for above program.
 - Use XML based configuration and perform autowiring with different types. (byName, byType and constructor). Use one autowiring type at a time.
- 4) Example on @Controller, @Service, @Repository, @Autowired, @Configuration and @Bean
- Modify the above application, use annotations and java based configuration.
- 5) Write a program to demonstrate use of @Resource, @Inject, @Required annotations
- 6) Example of @Component, @Value, @PropertySource & Environment
- a. Create a dbConfig.properties file which contains database configuration details like driver class name, dburl, username, password.

- b. Create a Java class in which you have to read all properties and display on a console. (Use `@Component`, `@Value` or `Environment` and `@PropertyResource`).
-

- 7) Write a Java program to demonstrate SPEL (Spring Expression language)
- 8) Write a Java program to demonstrate `InitializingBean` and `DisposableBean`.
Try Different ways:
(Use `init-method` and `destroy-method` in xml config file)
(Use `@PostConstruct` and `@PreDestroy`)

9) Write a Java program to demonstrate Complete Bean Life cycle.

10) Write a java program to demonstrate ApplicationContextAware interface.