[DBConfig-SetterBasedInjection-VU](https://cognizant.tekstac.com/mod/vpl/view.php?id=48695)

**Grade settings**: Maximum Grade: 100  
**Disable external file upload, paste and drop external content**: Yes  
**Based on**: [DBConfig-SetterBasedInjection-VU](https://cognizant.tekstac.com/mod/vpl/view.php?id=48695)  
**Run**: Yes **Save**: Yes **Evaluate**: Yes  
**Automatic grade**: Yes **Maximum execution time**: 32 s **Maximum memory used**: 1.50 GiB

**Objective:**

To work with a Spring Core application using annotations, bean configuration, setter-based injection, and Spring JDBC concepts.

**Concept Explanation:**

1. **Setter-based Injection:** Dependencies are injected into beans using setter methods, allowing for flexibility and easy reconfiguration of dependencies.
2. **Spring JDBC**simplifies database access by providing utilities for connecting to databases, executing SQL queries, and handling exceptions, reducing boilerplate code and promoting abstraction.

**Code Implementation:** 

1. **Setter-based injection** is utilized to inject values into the attributes of the **DBConfig** and **StudentDAO** classes.
2. This is achieved by providing setter methods for the attributes and using annotations like **@Value** to inject values into these setters.
3. The inclusion of a **DBConfig** class suggests the intention to configure database connection details.
4. This aligns with Spring JDBC concepts, where Spring simplifies database access by providing utilities for connecting to databases, executing SQL queries, and handling exceptions.

Create a class called DBconfig  with the below attributes

|  |  |
| --- | --- |
| driverName | String |
| url | String |
| userName | String |
| password | String |

Include getters and setters for all the above mentioned attributes.

Create a class called **ApplicationConfig** that has the required annotations for **scanning** and **registering** the bean definitions.

The attributes inside the DBConfig should be configured with the below values

|  |  |
| --- | --- |
| Attribute Name | Value |
| driverName | oracle.jdbc.driver.OracleDriver |
| url | jdbc:oracle:thin:@localhost:1521:oracle |
| userName | alex |
| password | alex@123 |

The values for these attributes should be injected via annotations in the dbConfig() method in ApplicationConfig class. Provide the necessary annotation above the dbConfig() method to register that as a bean.

Create class called StudentDAO that has below attribute

|  |  |
| --- | --- |
| dbConfig | DBConfig |

Include getter/setter for the above attribute

The values for these attributes should be injected via annotations in the studentDAO() method in ApplicationConfig class. Provide the necessary annotation above the studentDAO() method to register that as a bean.

Create a class called Driver with the main method. Retrieve the StudentDAO object and display the details as shown in the sample input/output.

**Design Constraints:**

DBConfig class should be present in com.spring.app package

StudentDAO class should be present in com.spring.app package

Main class should be present in com.app.driver package

DBConfig and StudentDAO classes should have the necessary getters and setters.

The className/Attribute Name/PackageName should be the same as specified in the problem statement. Do not create any new packages.

Do not create any XML configuration file. You need to use only annotations for configuration.

**Sample Input/Output:**

oracle.jdbc.driver.OracleDriver

jdbc:oracle:thin:@localhost:1521:oracle

alex

alex@123

[Next activity](https://cognizant.tekstac.com/mod/vpl/view.php?id=20536&forceview=1)**[Azure Horizon Resort](https://cognizant.tekstac.com/mod/vpl/view.php?id=20536&forceview=1)**

**Kudos!**You have earned some XP points.