

Table of Contents

Step 1: Prepare model and entity	2
Step 2: Database configuration in spring project	2
Step 3: Inject JdbcTemplate.....	2
Step 4: Perform Crud Operations	3
Step 4: Properties file and @PropertySource annotation.....	4

BELOW CODE IS TO BE ADDED IN SpringCoreDemo project created

Step 1: Prepare model and entity

1. Create table book id, name and price as columns. Id as primary key
2. Insert few records in database for book table
3. Create a class Book as follows:

```
public class Book {  
    private int id;  
    private String name;  
    private double price;  
    // getters and setters  
    // constructor  
    // toString  
}
```

Step 2: Database configuration in spring project

4. Create a class JavabasedConfiguration within package com.demo.jdbc and write below code:

```
@Configuration  
public class JavabasedConfiguration {  
  
    @Bean  
    public DataSource getDataSource()  
    {  
        DriverManagerDataSource dataSource = new DriverManagerDataSource();  
        dataSource.setDriverClassName("com.mysql.jdbc.Driver");  
        dataSource.setUrl("jdbc:mysql://localhost:8889/java");  
        dataSource.setUsername("root");  
        dataSource.setPassword("root");  
        System.out.println("Data source created");  
        return dataSource;  
    }  
  
    @Bean  
    @Autowired  
    public JdbcTemplate getTempalte(DataSource ds){  
        return new JdbcTemplate(ds);  
    }  
}
```

Step 3: Inject JdbcTemplate

1. Create a class BookDatabase with a dependency on JdbcTemplate as follows :
@Repository annotation is a semantic annotation for @Component

```

@Repository
public class BookDatabase {

    JdbcTemplate jdbcTemplate;

    public void setJdbcTemplate() {
        return this.jdbcTemplate = jdbcTemplate;
    }

}

```

2. Now in App class, call the getJdbcTemplate method and check if it is null or not.
3. **If not null then proceed to next step**

Step 4: Perform Crud Operations

1. Update BookDatabase as follows :

```

public int getBookCount()
{
    int count =this.jdbcTemplate.queryForObject("select count(*) from book",
Integer.class);
    return count;
}
public void insertBook(Book book) {
    // Line 1
    String sql = "insert into Book values(?,?,?)";
    // Line 2
    int count = jdbcTemplate.update(sql,
book.getId(),Book.getName(),Book.getPrice());
    // Line 3
    System.out.println("====>>> count :: " + count);
}
public void updateBook(Integer id, String name) {
    // Line 1
    String sql = "update book set name=? where id = ? ";
    // Line 2
    int count = jdbcTemplate.update(sql, name, id);
    // Line 3
    System.out.println("====>>> count :: " + count);
}
public void deleteBook(Integer id) {
    // Line 1
    String sql = "delete from book where id = ? ";
    // Line 2
    int count = jdbcTemplate.update(sql,id);
    // Line 3
    System.out.println("====>>> count :: " + count);
}
}

```

```

public void queryForBookName(Integer id) {
    String sql = "SELECT name FROM Book where id = ?";
    String name = jdbcTemplate.queryForObject(sql,
        new Object[]{id},String.class);
    System.out.println("====>>> Name :: " + name);
}
public void queryForBook(Integer id) {
    String sql = "SELECT * FROM book where id = ?";
    Book Book = jdbcTemplate.queryForObject(sql,
        new Object[] {id}, new BookMapper());
    System.out.println("====>>> Book details :: " + Book);
}
public void queryForBooks() {
    String sql = "SELECT * FROM book";
    List<Book> Book = jdbcTemplate.query(sql, new BookMapper());
    System.out.println("ID\tName\tPrice");
    for (Book book : Book)
    {
        System.out.print(book.getId()+"\t");
        System.out.print(book.getName()+"\t");
        System.out.print(book.getPrice()+"\t");
        System.out.println();
    }
}

// make sure to correctly import the RowMapper
// import org.springframework.jdbc.core.RowMapper;

class BookMapper implements RowMapper<Book>{
    public Book mapRow(ResultSet rs, int rowNum) throws SQLException {
        Book c1 = new Book();
        c1.setId(rs.getInt(1));
        c1.setName(rs.getString(2));
        c1.setPrice(rs.getDouble(3));
        return c1;
    }
}

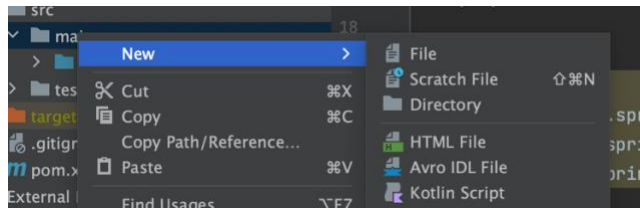
```

Step 4: Properties file and @PropertySource annotation

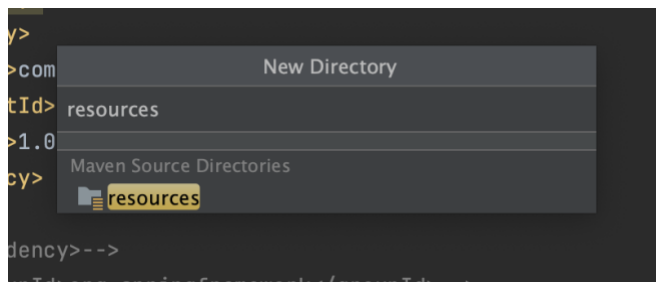
1. Hardcoding the database credentials is not a good practice.
2. Skip step 2 if resources folder is already there within main folder

3. Create resources folder [IF NOT ALREADY THERE] within the SpringCoreDemo project as follows:

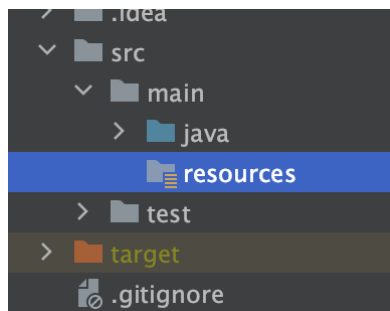
Right click main folder -> New -> Directory



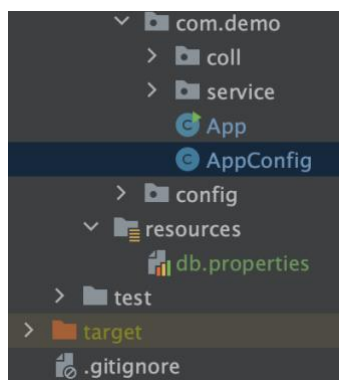
Type resources as directory name



It looks as below



Create a file db.properties within resources folder



4. Update the properties file as follows:

```
url=jdbc:mysql://localhost:8889/jdbc  
username=root
```

```
password=root  
driver=com.mysql.cj.jdbc.Driver
```

5. Update JavabasedConfiguration to read values from properties file as follows:

```
@Configuration  
@PropertySource("classpath:db.properties")  
public class JavabasedConfiguration {  
  
    @Value("${driver}")  
    private String driver;  
  
    @Value("${url}")  
    private String url;  
  
    @Value("${username}")  
    private String username;  
  
    @Value("${password}")  
    private String password;  
  
    @Bean  
    public DataSource getDataSource()  
    {  
        DriverManagerDataSource dataSource = new DriverManagerDataSource();  
        dataSource.setDriverClassName(driver);  
        dataSource.setUrl(url);  
        dataSource.setUsername(username);  
        dataSource.setPassword(password);  
        System.out.println("Data source created");  
        return dataSource;  
    }  
  
    ...  
}
```

Everything else should work the same