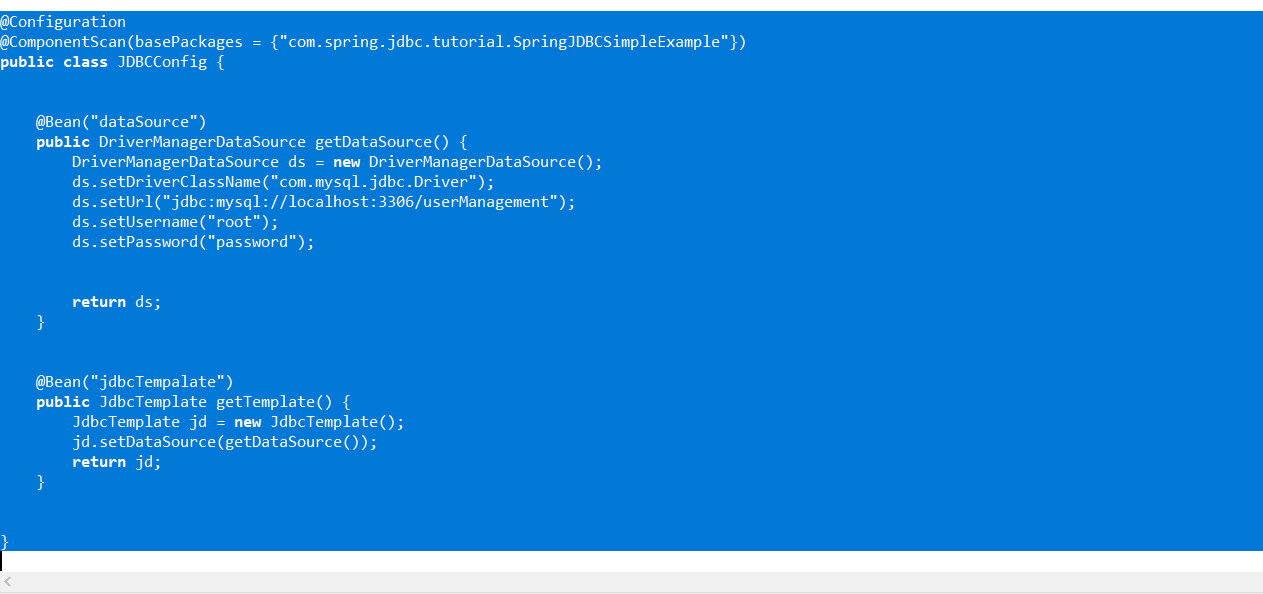
# Q1.What is Spring JDBC?

* Spring JDBC was introduced to overcome the disadvantages of traditional JDBC
* Following are some of the disadvantages of JDBC
  + We have to repeate the same code again and again for one database operation. I.e open the connection,prepare statement,execute, close the connection
  + In traditional JDBC we have to handle Exception such as SQL Exception
  + Since we are repeating the code it take a lot of developer time
* Spring JDBC use JDBCTemplate class to provide all the database operations and as a developer we do not have to open or close the connection

# Q2.How to establish a connection to database from JDBC?

* In Spring JDBC, we will create the object of JDBCTempalate Class and we need to set the datasource inside JDBCTempalate class
* So we also need to create the object of Datasource class. In Datasource object we will set the url,username,password,driver etc..
* After creating the datasource object we will use setDataSourceMethod of JDBCTempalate to establish the connection
* Since we are using spring we need to achieve dependency injection I.e we should be able to use object of tempalate class anywhere in the program so for that we can use JavaConfgiurationMethod. So for that we will have the class with @Configuration annoation for eg
* 
* We can get the object of JDBCTempalate using @Autowired annotation

# Q3.How to insert,update or delete the record using Spring JDBC?

* We will create the string having our sql query as we used to have in jdbc
* We will have insert,update and delete query as per the requirement
* We will use update() method of jdbcTemplate which we will get using dependency injection and passs our string query and parameters

String query="delete from student where rollno = ?";

**int** i = **this**.jdbcTemplate.update(query,s.getRollNumber());

# Q4.How to Select the data usin Spring JDBC?

* Spring JDBC provide us the object or list of object of particular class instead of providing the resultSet.
* For this a developer has to implement getMapRow() method of RowMapper Interface. This method will internally convert the resultSet into object
* We have to either pass the object of RowMapper class which implements getMapRow() or we can create the anonymous class

**public** Student getStudent(**int** rollNumber) {

String sql= "select rollno,name,marks from student where rollno = ?";

Student s = **this**.jdbcTemplate.queryForObject(sql, **new** RowMapper<Student>() {

@Override

**public** Student mapRow(ResultSet rs, **int** rowNum) **throws** SQLException {

// **TODO** Auto-generated method stub

Student s = **new** Student();

s.setMarks(rs.getInt(3));

s.setName(rs.getString(2));

s.setRollNumber(rs.getInt(1));

**return** s;

}

},rollNumber);

**return** s;

}

* If there are multiple rows the return type will be List and we will use query() method of jdbcTempalate. For single row we will use queryForObject() method. The implementaion of mapRow will remain same