

Lesson 5

[Fubar-Cyber/lesson5: Hibernate with MySQL. You will need to make sure you have MySQL installed and create a Database and Table\(s\) to be used to save data from your application. \(github.com\)](https://github.com/Fubar-Cyber/lesson5)

ToDoList.java

```
package org.hibernate.lesson5;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

import java.time.LocalDateTime;
import java.util.List;
import java.util.Scanner;

public class ToDoList {
    private SessionFactory sessionFactory;

    public ToDoList() {
        sessionFactory = new Configuration()
            .configure("hibernate.cfg.xml")
            .buildSessionFactory();
    }

    public void addItem(String item) {
        ToDoItem toDoItem = new ToDoItem();
        toDoItem.setDescription(item);
        toDoItem.setCreatedAt(LocalDateTime.now());

        try (Session session = sessionFactory.openSession()) {
            session.beginTransaction();
            session.save(toDoItem);
            session.getTransaction().commit();
        }
    }

    public void deleteItem(int id) {
        try (Session session = sessionFactory.openSession()) {
            session.beginTransaction();
            ToDoItem toDoItem = session.get(ToDoItem.class, id);
            if (toDoItem != null) {
                session.delete(toDoItem);
                session.getTransaction().commit();
            } else {
                System.out.println("Item not found.");
            }
        }
    }

    public void displayList() {
```

```

        try (Session session = sessionFactory.openSession()) {
            List<ToDoItem> itemList = session.createQuery("FROM ToDoItem",
ToDoItem.class).getResultList();
            if (itemList.isEmpty()) {
                System.out.println("Your To-Do list is empty! Go relax and
have fun!");
            } else {
                for (ToDoItem item : itemList) {
                    System.out.println(item.getId() + ". " +
item.getDescription());
                }
            }
        }
    }

    public static void main(String[] args) {
        ToDoList todoList = new ToDoList();
        Scanner scanner = new Scanner(System.in);
        while (true) {
            System.out.println("What would you like to do? (add, delete,
view, quit):");
            String command = scanner.nextLine().toLowerCase(); // Convert
input to lowercase
            if (command.equalsIgnoreCase("add")) {
                System.out.println("Enter an item to add:");
                String item = scanner.nextLine();
                todoList.addItem(item);
            } else if (command.equalsIgnoreCase("delete")) {
                System.out.println("Enter the ID of the item to delete:");
                int id = scanner.nextInt();
                scanner.nextLine(); // Consume the newline character
                todoList.deleteItem(id);
            } else if (command.equalsIgnoreCase("view")) {
                System.out.println("The items on your To-Do List are:\n");
                todoList.displayList();
            } else if (command.equalsIgnoreCase("quit")) {
                System.out.println("Goodbye! Have a great day!");
                break;
            } else {
                System.out.println("Invalid command!");
            }
        }
        scanner.close();
    }
}

```

ToDoItem.java

```

package org.hibernate.lesson5;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

```

```

import java.time.LocalDateTime;

@Entity
@Table(name = "todoitem")
public class ToDoItem {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    @Column(name = "description")
    private String description;

    @Column(name = "created_at")
    private LocalDateTime createdAt;

    public ToDoItem() {
        // Default constructor required by Hibernate
    }

    public int getId() {
        return id;
    }

    public String getDescription() {
        return description;
    }

    public void setDescription(String description) {
        this.description = description;
    }

    public LocalDateTime getCreatedAt() {
        return createdAt;
    }

    public void setCreatedAt(LocalDateTime createdAt) {
        this.createdAt = createdAt;
    }
}

```

Hibernate.cfg.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
    "-//Hibernate/Hibernate Configuration DTD/EN"
    "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>
        <!-- Database connection properties -->
        <property
name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>
        <property
name="hibernate.connection.driver_class">com.mysql.cj.jdbc.Driver</property>
        <property
name="hibernate.connection.url">jdbc:mysql://localhost:3306/databasetodolist<
/property>
    
```

```

        <property name="hibernate.connection.username">root</property>
    </property>
    name="hibernate.connection.password">MySQLPassword</property>

    <!-- Entity class mapping -->
    <mapping class="org.hibernate.lesson5.ToDoItem"/>
</session-factory>
</hibernate-configuration>

```

Lesson5/pom.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>org.hibernate.lesson5</groupId>
    <artifactId>your-project</artifactId>
    <version>1.0-SNAPSHOT</version>

    <properties>
        <maven.compiler.source>1.8</maven.compiler.source>
        <maven.compiler.target>1.8</maven.compiler.target>
    </properties>

    <dependencies>
        <!-- Hibernate dependencies -->
        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-core</artifactId>
            <version>5.5.7.Final</version>
        </dependency>

        <!-- JPA dependency -->
        <dependency>
            <groupId>javax.persistence</groupId>
            <artifactId>javax.persistence-api</artifactId>
            <version>2.2</version>
        </dependency>

        <!-- JUnit dependency -->
        <dependency>
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
            <version>4.13.1</version>
            <scope>test</scope>
        </dependency>

        <!-- MySQL Connector/J dependency -->
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>8.0.26</version>

```

```

        </dependency>

        <!-- Other dependencies -->
        <!-- ... -->
    </dependencies>

    <build>
        <plugins>
            <!-- Maven Compiler Plugin -->
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.8.1</version>
                <configuration>
                    <source>1.8</source>
                    <target>1.8</target>
                </configuration>
            </plugin>
        </plugins>
    </build>
</project>

```

MySQLConnectionTest.java

```

package org.hibernate.lesson5;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class MySQLConnectionTest {
    public static void main(String[] args) {
        String jdbcUrl = "jdbc:mysql://localhost:3306/databasetodolist";
        String username = "root";
        String password = "MySQLPassword";

        try (Connection connection = DriverManager.getConnection(jdbcUrl,
username, password)) {
            System.out.println("Connection successful! You are connected to
the MySQL database.");
        } catch (SQLException e) {
            System.out.println("Connection failed. Error message: " +
e.getMessage());
        }
    }
}

```

```
5/ToDoList.java - Eclipse IDE
ject Run Window Help

1 package org.hibernate.lesson5;
2
3 import org.hibernate.Session;
4 import org.hibernate.SessionFactory;
5 import org.hibernate.cfg.Configuration;
6
7 import java.time.LocalDateTime;
8 import java.util.List;
9 import java.util.Scanner;
10
11 public class ToDoList {
12     private SessionFactory sessionFactory;
13
14     public ToDoList() {
15         sessionFactory = new Configuration()
16             .configure("hibernate.cfg.xml")
17             .buildSessionFactory();
18     }
19
20     public void addItem(String item) {
21         ToDoItem toDoItem = new ToDoItem();
22         toDoItem.setDescription(item);
23         toDoItem.setCreatedAt(LocalDateTime.now());
24
25         try (Session session = sessionFactory.openSession()) {
26             session.beginTransaction();
27             session.save(toDoItem);
28         }
29     }
30 }

Markers Properties Servers Data Source Explorer Snippets Terminal Console Package Explorer
ToDoList (2) [Java Application] C:\Users\Jensy Fernandez\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_19.0.2.v20230129-1123\jre\bin\javaw.exe (Jul
What would you like to do? (add, delete, view, quit):
add
Enter an item to add:
Lesson 1
What would you like to do? (add, delete, view, quit):
add
Enter an item to add:
Lesson2
What would you like to do? (add, delete, view, quit):
Lesson 3
Invalid command!
What would you like to do? (add, delete, view, quit):
add
Enter an item to add:
Lesson 3
What would you like to do? (add, delete, view, quit):
add
Enter an item to add:
Lesson 4
What would you like to do? (add, delete, view, quit):
add
Enter an item to add:
Lesson 5
What would you like to do? (add, delete, view, quit):
```

```

6
7 import java.time.LocalDateTime;
8 import java.util.List;
9 import java.util.Scanner;
10
11 public class ToDoList {
12     private SessionFactory sessionFactory;
13
14     public ToDoList() {
15         sessionFactory = new Configuration()
16             .configure("hibernate.cfg.xml")
17             .buildSessionFactory();
18     }
19
20     public void addItem(String item) {
21         ToDoItem toDoItem = new ToDoItem();
22         toDoItem.setDescription(item);
23         toDoItem.setCreatedAt(LocalDateTime.now());
24     }

```

Markers Properties Servers Data Source Explorer Snippets Terminal Console × Package Explorer

ToDoList (2) [Java Application] C:\Users\Jensy Fernandez\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_19.0.2.v20230129-1123\jre\bin\java

Enter an item to add:

Lesson 5

What would you like to do? (add, delete, view, quit):

view

The items on your To-Do List are:

4. Lesson 1

5. Lesson2

6. Lesson 3

7. Lesson 4

8. Lesson 5

What would you like to do? (add, delete, view, quit):

delete

Enter the ID of the item to delete:

8

What would you like to do? (add, delete, view, quit):

delete

Enter the ID of the item to delete:

7

What would you like to do? (add, delete, view, quit):

delete

Enter the ID of the item to delete:

6

What would you like to do? (add, delete, view, quit):

delete

Enter the ID of the item to delete:

5

What would you like to do? (add, delete, view, quit):

The screenshot displays the DBeaver IDE interface. On the left, the 'HEMAS' database structure is visible, showing tables 'todoitem' and 'todolist'. The main editor window shows a SQL query: `SELECT * FROM databasetodolist.todoitem;`. The bottom panel shows the 'Output' window with a table of execution results. The table has columns: #, Time, Action, and Message. The messages indicate the number of rows returned for each query execution.

#	Time	Action	Message
12	18:33:05	SELECT * FROM databasetodolist.todoitem LIMIT 0, 1000	2 row(s) returned
13	18:34:30	SELECT * FROM databasetodolist.todoitem LIMIT 0, 1000	2 row(s) returned
14	18:36:05	SELECT * FROM databasetodolist.todoitem LIMIT 0, 1000	0 row(s) returned
15	19:25:44	SELECT * FROM databasetodolist.todoitem LIMIT 0, 1000	5 row(s) returned
16	19:28:20	SELECT * FROM databasetodolist.todoitem LIMIT 0, 1000	1 row(s) returned
17	19:29:07	SELECT * FROM databasetodolist.todoitem LIMIT 0, 1000	0 row(s) returned


```

14 public ToDoList() {
15     sessionFactory = new Configuration()
16         .configure("hibernate.cfg.xml")
17         .buildSessionFactory();
18 }
19
20 public void addItem(String item) {
21     ToDoItem toDoItem = new ToDoItem();
22     toDoItem.setDescription(item);
23     toDoItem.setCreatedAt(LocalDateTime.now());
24 }

```

Markers Properties Servers Data Source Explorer Snippets Terminal Console Package Explorer

ToDoList (2) [Java Application] C:\Users\Jensy Fernandez\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_19.0.2.v2023012

Enter the ID of the item to delete:

8

What would you like to do? (add, delete, view, quit):

delete

Enter the ID of the item to delete:

7

What would you like to do? (add, delete, view, quit):

delete

Enter the ID of the item to delete:

6

What would you like to do? (add, delete, view, quit):

delete

Enter the ID of the item to delete:

5

What would you like to do? (add, delete, view, quit):

Invalid command!

What would you like to do? (add, delete, view, quit):

delete

Enter the ID of the item to delete:

4

What would you like to do? (add, delete, view, quit):

view

The items on your To-Do List are:

Your To-Do list is empty! Go relax and have fun!

What would you like to do? (add, delete, view, quit):

Edit View Query Database Server Tools Scripting Help

Navigator: Filter objects

database:todoist

Tables

todoitem

Views

Stored Procedures

Functions

sys

todoist

ministration Schemas

ormation

Table: todoitem

Columns:

id int AI PK

description varchar(45)

created_at varchar(45)

todoitem todoitem - Table todoitem - Table todoitem todoitem todoitem todoitem todoitem todoitem

Limit to 1000 rows

1 • SELECT * FROM database:todoist.todoitem;

Result Grid

	id	description	created_at
▶	4	Lesson 1	2023-07-08 19:23:18.464334
	5	Lesson2	2023-07-08 19:23:28.144928
	6	Lesson 3	2023-07-08 19:23:53.702553
	7	Lesson 4	2023-07-08 19:24:03.620236
	8	Lesson 5	2023-07-08 19:24:12.430961
•	NULL	NULL	NULL

todoitem 1 x

Output

Action Output

Result Grid

Form Editor

Field Types

Context Hel