

Soumyajit Mondal - 23262325

TELECOM NMS - JAVA

DAY 1

Generic Class

```
1 package Java_AdvanceTraining_WIPRO;
2
3 public class Day1_Generic_Class {
4
5     int id;
6     String name;
7     String dept;
8     String designation;    I
9
10    public Day1_Generic_Class (int id, String name, String dept,
11                           String designation) {
12        this.id = id;
13        this.name = name;
14        this.dept = dept;
15        this.designation = designation;
16    }
17
18
19    @Override
20    public String toString() {           I
21        return "Day1_Generic_Class[id =" + id + ", name =" + name + ","
22               + " dept =" + dept + ", designation =" + designation + "]";
23    }
24
25    class Generics1 <X>{
26        X data;
27        void assign (X data) {
28            this.data = data;
29        }
30
31        void display() {
32            System.out.println("Given data : " + data);
33        }
34    }
```

```
35④ public static void main(String[] args) {
36     // TODO Auto-generated method stub
37     Generics1 <Integer> intObj = new Generics1 <Integer>();
38     intObj.assign(100);
39     intObj.display();
40
41     Generics1 <String> StrObj = new Generics1 <String>();
42     StrObj.assign("Java Generics");
43     StrObj.display();
44
45     Day1_Generic_Class empObj = new Day1_Generic_Class (101,
46             "Soumyajit", "IT", "Full Stack Developer");
47     Generics1<Day1_Generic_Class> empGen =
48             new Generics1<Day1_Generic_Class>();
49     empGen.assign(empObj);
50     empGen.display();
51
52 }
53
54 }
```

Problems Javadoc Declaration Console ×
<terminated> Generics1 [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (19-Apr-2024, 2:41:47 pm – 2:41:47 pm) [pid: 4696]
Given data : 100
Given data : Java Generics
Given data : Day1_Generic_Class[id =101, name =Soumyajit, dept =IT, designation=Full Stack Developer]

Generic Methods

```
1 package Java_AdvanceTraining_WIPRO;
2
3 public class Day1_Generic_Methods {
4
5     public static <E> void printArray (E[] inputArray) {
6
7         System.out.println();
8         for (E el: inputArray)
9             System.out.print(el + ",");
10    }
11}
```

```

11
12  public static void main(String[] args) {
13      // TODO Auto-generated method stub
14
15      Integer[] intArray = { 1,2,3,4,5};
16      Double[] doubleArray = {1.1, 1.2, 1.3, 1.4, 1.5};
17      Character[] charArray = {'j','a','v','a'};
18
19      printArray(intArray); |
20      printArray(doubleArray);
21      printArray(charArray);
22  }
23
24 }
25

```

Problems @ Javadoc Declaration Console <terminated> Day1_Generic_Methods [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (19-Apr-2024, 2:46:28 pm – 2:46:28 pm) [pid: 2864]

1,2,3,4,5,
1.1,1.2,1.3,1.4,1.5,
j,a,v,a,

Activate Windows

Java Reflection

Reflection Class Information

```

1 package Java_AdvanceTraining_WIPRO;
2
3 import java.lang.reflect.Modifier;
4
5 class Employee2{
6
7 }
8

```

```

9 public class Day1_Reflection_Class {
10
11    public static void main(String[] args) {
12        // TODO Auto-generated method stub
13        Employee2 emp = new Employee2();
14        Class obj = emp.getClass();
15
16        String className = obj.getName();
17        System.out.println("Name of the class is : " + className);
18
19        int intModifiers = obj.getModifiers();
20
21        String strMod = Modifier.toString(intModifiers);
22        System.out.println("Modifiers of the class : "+strMod);
23
24        Class superClass = obj.getSuperclass();
25        System.out.println("Super class of Employee2 is : "+
26            superClass.getName());
27    }
28}

```

The screenshot shows an IDE interface with a code editor and a terminal window. The code editor contains the Java code provided above. The terminal window displays the following output:

```

Problems @ Javadoc Declaration Console X
<terminated> Day1_Reflection_Class [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (19-Apr-2024, 2:51:43 pm – 2:51:43 pm) [pid: 7916]
Name of the class is : Java_AdvanceTraining_WIPRO.Employee2
Modifiers of the class :
Super class of Employee2 is : java.lang.Object

```

Reflection Method Information

```

1 package Java_AdvanceTraining_WIPRO;
2
3 import java.lang.reflect.Method;
4
5
6 class Employee3 {
7     public String getData() {
8         return "Returning the data";
9     }
10
11    public void setData() {
12
13    }
14
15 }
16
17 public class Day1_Refelction_Method {
18
19    public static void main(String[] args) {
20        // TODO Auto-generated method stub
21

```

```

19④ public static void main(String[] args) {
20     // TODO Auto-generated method stub
21
22     Employee3 emp = new Employee3();
23     Class obj = emp.getClass();
24
25     Method listOfMethods[] = obj.getDeclaredMethods();
26     for (Method m: listOfMethods) {
27         String methodName = m.getName();
28         System.out.println("\nMethod name is :" +methodName);
29
30         int intModifier = m.getModifiers();
31         String strModifier = Modifier.toString(intModifier);
32         System.out.println("Modifier of the class : " + strModifier);
33         System.out.println("Method return type : " + m.getReturnType());
34     }
35
36 }
37
38 }

```

Line: 33

The screenshot shows an IDE interface with the following details:

- Toolbar:** Task List, Problems, Javadoc, Declaration, Console.
- Console Tab:** Active tab.
- Console Output:**

```

<terminated> Day1_Refelction_Method [Java Application] C:\Program Files\Java\jdk-21\bin
|
Method name is :setData
Modifier of the class : public
Method return type : void

Method name is :getData
Modifier of the class : public
Method return type : class java.lang.String

```

Reflection Accessing Private Attribute

```

1 package Java_AdvanceTraining_WIPRO;
2
3④ import java.lang.reflect.Field;□
4
5
6 class Employee4{
7     private String name;
8 }
9
10
11 public class Day1_Reflection_PrivateAttribute {

```

```
11 public class Day1_Reflection_PrivateAttribute {  
12  
13     public static void main(String[] args) throws NoSuchFieldException,  
14             SecurityException, IllegalArgumentException, IllegalAccessException {  
15         // TODO Auto-generated method stub  
16  
17         Employee4 emp = new Employee4();  
18         Class obj = emp.getClass();  
19         Field field1 = obj.getDeclaredField("name");  
20         field1.setAccessible(true);  
21         field1.set(emp, "Soumya");  
22         |  
23         String nameData = (String) field1.get(emp);  
24         System.out.println("\nPrivate data modified...Name value : "  
25             +nameData);  
26         int modifier1 = field1.getModifiers();  
27         String strModifier1 = Modifier.toString(modifier1);  
28         System.out.println("Field Modifier is : " + strModifier1);  
29  
30     }  
31  
32 }
```

The screenshot shows an IDE interface with a code editor and a console window. The code editor displays the Java code provided above. The console window is active and shows the output of the program's execution. The output consists of two lines of text: 'Private data modified...Name value : Soumya' and 'Field Modifier is : private'. The console tab is highlighted in blue, and there are various icons and buttons typical of an IDE's toolbar.

```
<terminated> Day1_Reflection_PrivateAttribute [Java Application] C:\Program Files\Java\jre1.8.0_251\bin\javaw.exe  
|  
| Private data modified...Name value : Soumya  
| Field Modifier is : private
```

Lambda Expression

Lambda Expression

The screenshot shows a Java code editor and a terminal window. The code editor displays a file named Day1_LambdaExpression.java with the following content:

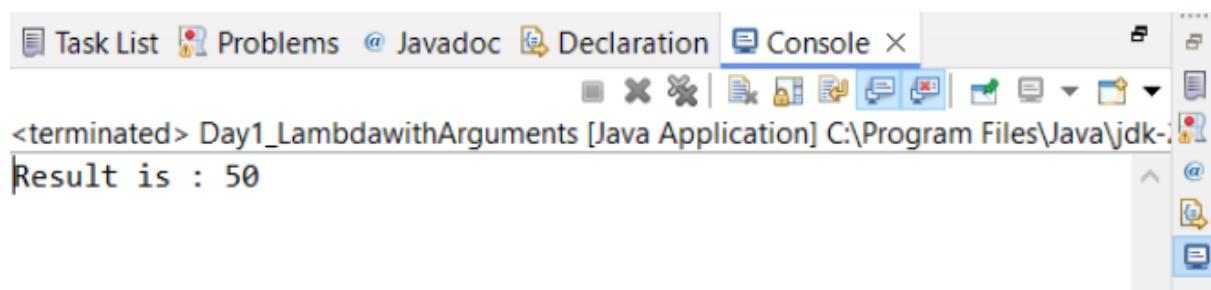
```
1 package Java_AdvanceTraining_WIPRO;
2
3 interface ITest{
4     public void display();
5 }
6
7 public class Day1_LambdaExpression {
8
9     public static void main(String[] args) {
10         // TODO Auto-generated method stub
11
12         ITest t1 = () -> System.out.println("Displaying information "
13             + "in Lambda....");
14         t1.display();
15     }
16
17 }
18
19 }
```

The terminal window below the code editor shows the output of the program:

```
<terminated> Day1_LambdaExpression [Java Application] C:\Program Files\Java\jdk-21\bin
Displaying information in Lambda....
```

Lambda With Arguments

```
1 package Java_AdvanceTraining_WIPRO;
2
3 interface IMaths{
4     public int sum(int x, int y);
5 }
6
7 public class Day1_LambdawithArguments {
8
9     public static void main(String[] args) {
10         // TODO Auto-generated method stub
11
12         IMaths m1 = (x,y) -> x + y;
13         int x = 20 , y = 30;
14         int result = m1.sum(x, y);
15         System.out.println("Result is : " + result); |
16
17     }
18
19 }
20
```



Predicate

Predicate

```
1 package Java_AdvanceTraining_WIPRO;
2
3 import java.util.Arrays;
4
5
6 public class Day1_Predicate {
7
8     public static void main(String[] args) {
9         // TODO Auto-generated method stub
10
11         List<Integer> numLst = Arrays.asList(10, 11, 12, 13, 14, 15,
12             16, 17, 18, 19, 20);
13         System.out.println("Original List...");
14         System.out.println(numLst);
15
16         Predicate<Integer> p1 = data -> data %2 == 0;
17
18         List filteredList = numLst.
19             stream().
20             filter(p1).
21             collect(Collectors.toList());
22         System.out.println("\nFiltered List...");
23         System.out.println(filteredList);
24     }
25
26 }
27
28 }
```

The screenshot shows an IDE interface with a code editor and a console window. The code editor contains the Java code provided above. The console window shows the execution of the program:

```
<terminated> Day1_Predicate [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe
Original List...
[10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]

Filtered List...
[10, 12, 14, 16, 18, 20]
```

Method Referencing

```
1 package Java_AdvanceTraining_WIPRO;
2
3 interface INewMap{
4     public void displayDirections();
5 }
6
7 public class Day1_Method_Refencing {
8
9     Day1_Method_Refencing(){
10         System.out.println("Constructor of the class Test...");
11     }
12     public void displayGoogleMap() {
13         System.out.println("Displaying google map from src to destination..");
14     }
15     public static void displayGoogleMapsStatic() {
16         System.out.println("Displaying google maps from a static method..");
17     }
18
19     public static void main(String[] args) {
20         // TODO Auto-generated method stub
21
22         Day1_Method_Refencing Day1_Method_RefencingObj =
23             new Day1_Method_Refencing();
24         INewMap IN = Day1_Method_RefencingObj :: displayGoogleMap;
25         IN.displayDirections();
26
27         INewMap InStatic = Day1_Method_Refencing::displayGoogleMapsStatic;
28         InStatic.displayDirections();
29
30         INewMap INCons = Day1_Method_Refencing :: new;
31         INCons.displayDirections();
32
33     }
34
35 }
```

The screenshot shows an IDE interface with the following details:

- Code Editor:** Displays the Java code for `Day1_Method_Refencing`. The code defines an interface `INewMap` with a method `displayDirections()`. It also contains three methods: `displayGoogleMap()`, `displayGoogleMapsStatic()`, and the `main` method which demonstrates various ways to call these methods using method references.
- Console Tab:** Labeled "Console X".
- Console Output:** The output shows the execution of the `main` method. It prints:
 - "Constructor of the class Test..."
 - "Displaying google map from src to destination.."
 - "Displaying google maps from a static method.."
 - "Constructor of the class Test..."

DAY 2

File Operations

FileWriter

```
1 package Java_AdvanceTraining_Day2_Wipro;
2
3 import java.io.FileWriter;
4
5 public class FileOperations_FileWriter {
6
7     public static void main(String[] args) throws IOException {
8         // TODO Auto-generated method stub
9
10        FileWriter fw = new FileWriter("C:\\TELECOM NMS JAVA");
11
12        fw.write("hello, welcome to java..");
13        System.out.println("Data writting into a file successfully..");
14
15        fw.close();
16    }
17
18}
19
20
```

FileReader

```
1 package Java_AdvanceTraining_Day2_Wipro;
2
3 import java.io.FileNotFoundException;
4
5 public class FileOperations_FileReader {
6
7     public static void main(String[] args) {
8         // TODO Auto-generated method stub
9
10        try {
11            FileReader fr = new FileReader ("C:\\\\TELECOM NMS JAVA\\\\FileWriter.JAVA");
12
13            int i = 0;
14            while ((i = fr.read()) != -1) {
15                System.out.println((char)i);
16            }
17            fr.close();
18        } catch (FileNotFoundException e) {
19            e.printStackTrace();
20
21        } catch (IOException e) {
22            e.printStackTrace();
23        }
24
25    }
26
27}
28
```

Activate Windows

The screenshot shows a Java application window titled "FileOperations_FileReader [Java Application]". The console tab displays the following output:

```
W
r
i
t
i
n
g
u
s
i
n
g
I
b
u
f
f
e
r
s
i
n
t
o
i
n
t
o
a
f
i
l
e
.
.
```

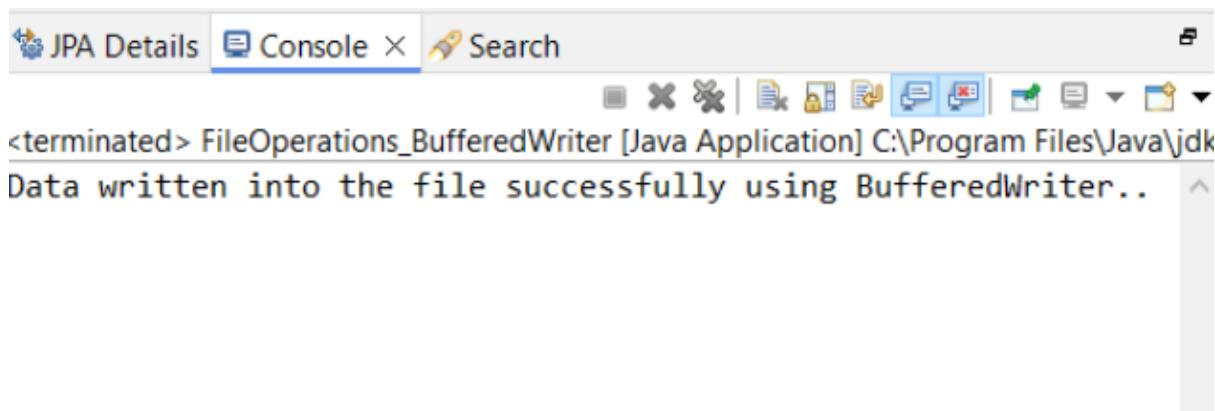
The code in the editor is:

```
1 package Java_AdvanceTraining_Day2_Wipro;
2
3 import java.io.BufferedReader;
4
5 public class FileOperations_BufferedWriter {
6
7     public static void main(String[] args) {
8         // TODO Auto-generated method stub
9
10        try {
11            FileWriter fw = new FileWriter("C:\\\\\\\\\\TELECOM NMS JAVA\\\\\\\\\\\\FileWriter.JAVA");
12            BufferedWriter buffer = new BufferedWriter (fw);
13            buffer.write("Writing using buffers into a file..");
14            System.out.println("Data written into the file successfully using BufferedWriter..");
15            buffer.close();
16
17        } catch (IOException e) {
18            e.printStackTrace();
19        }
20    }
21
22
23 }
24
25 }
```

A tooltip "Activate Windows" is visible near the bottom right of the screen.

BufferedWriter

```
1 package Java_AdvanceTraining_Day2_Wipro;
2
3 import java.io.BufferedReader;
4
5 public class FileOperations_BufferedWriter {
6
7     public static void main(String[] args) {
8         // TODO Auto-generated method stub
9
10        try {
11            FileWriter fw = new FileWriter("C:\\\\\\\\\\TELECOM NMS JAVA\\\\\\\\\\\\FileWriter.JAVA");
12            BufferedWriter buffer = new BufferedWriter (fw);
13            buffer.write("Writing using buffers into a file..");
14            System.out.println("Data written into the file successfully using BufferedWriter..");
15            buffer.close();
16
17        } catch (IOException e) {
18            e.printStackTrace();
19        }
20    }
21
22
23 }
24
25 }
```



BufferedReader

```
1 package Java_AdvanceTraining_Day2_Wipro;
2
3 import java.io.BufferedReader;[]
4
5 public class FileOperations_BufferedReader {
6
7     public static void main(String[] args) {
8         // TODO Auto-generated method stub
9
10        try {
11            FileReader fr = new FileReader("C:\\\\\\\\\\\\\\TELECOM NMS JAVA\\\\\\\\\\\\\\\\FileWriter.JAVA");
12            BufferedReader br = new BufferedReader(fr);
13            int i = 0;
14            while((i = br.read()) != -1) {
15                System.out.println((char)i);
16            }
17        } catch (FileNotFoundException e) {
18            e.printStackTrace();
19
20        } catch (IOException ex) {
21            System.out.println("Exception is :" + ex);
22        }
23    }
24
25 }
26
27 }
28
29 }
```

```
w
r
i
t
i
n
g
u
s
i
n
g
b
u
f
f
e
r
s
i
n
t
o
i
n
t
o
a
f
i
l
e
.
.
```

Serialization

```
1 package Java_AdvanceTraining_Day2_Wipro;
2
3 import java.io.FileNotFoundException;
4
5 class EmployeeSerialization implements Serializable{
6     int id;
7     String name;
8     String dept;
9     String designation;
10
11    EmployeeSerialization (int id, String name, String dept,
12                          String designation) {
13        this.id = id;
14        this.name = name;
15        this.dept = dept;
16        this.designation = designation;
17    }
18
19    public String toString() {
20        return "Employee Serialization [id=" + id + ", name=" + name + ", "
21               + "dept=" + dept + ","
22               + " designation =" + designation + "]";
23    }
24}
```

```
31 public class Serialization {  
32  
33     public static void main(String[] args) {  
34         // TODO Auto-generated method stub  
35         EmployeeSerialization empObj = new EmployeeSerialization  
36             (101, "Soumyajit", "IT", "Full Stack Developer");  
37         try {  
38             FileOutputStream fos = new FileOutputStream  
39                 ("C:\\SoumyaPractice.txt");  
40             ObjectOutputStream oos = new ObjectOutputStream (fos);  
41             oos.writeObject (empObj);  
42  
43             System.out.println("Employee Object is serialized successfully...");  
44             fos.close();  
45             oos.close();  
46  
47         }catch (FileNotFoundException ex) {  
48             System.out.println(ex);  
49  
50         }catch (IOException ex) {  
51             System.out.println(ex);  
52         }  
53     }  
54 }  
55  
56 }  
57 }
```

Activate Windows

The screenshot shows an IDE interface with the following details:

- Toolbar:** Includes tabs for JPA Details, Console (which is selected), and Search, along with various tool icons.
- Console Output:** Displays the following text:

```
<terminated> Serialization [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe  
Employee Object is serialized successfully...
```

Deserialization

```
1 package Java_AdvanceTraining_Day2_Wipro;
2
3* import java.io.FileInputStream;[]
4
5 class EmployeeSerialization implements Serializable{
6     int id;
7     String name;
8     String dept;
9     String designation;
10
11 EmployeeSerialization (int id, String name, String dept,
12                     String designation) {
13     this.id = id;
14     this.name = name;
15     this.dept = dept;
16     this.designation = designation;
17 }
18
19 public String toString() {
20     return "Employee Serialization [id=" + id + ", name=" + name + ", "
21         + "dept=" + dept + ","
22         + " designation =" + designation + "]";
23 }
24
25 public class Deserialization {
26
27     public static void main(String[] args) {
28         // TODO Auto-generated method stub
29
30         EmployeeSerialization empObj = new EmployeeSerialization
31             (101, "Soumyajit", "IT", "Full Stack Developer");
32         try {
33             FileInputStream fis = new FileInputStream
34                 ("C:\\SoumyaPractice.txt");
35
36             ObjectInputStream ois = new ObjectInputStream (fis);
37             EmployeeSerialization empobj =
38                 (EmployeeSerialization) ois.readObject();
39
40             System.out.println("Object deserialized is: " + empobj);
41
42         }catch (FileNotFoundException ex) {
43             System.out.println(ex);
44
45         } catch (IOException ex) {
46             System.out.println(ex);
47
48         } catch (ClassNotFoundException ex) {
49             System.out.println(ex);
50
51         }
52
53     }
54
55 }
56
57 }
58
59 }
```

Activate Windows

```
JPA Details Console X Search
<terminated> Deserialization [Java Application] C:\\Program Files\\Java\\jdk-21\\bin\\javaw.exe (08-May-2024, 3:52:17 pm - 3:52:20 pm) [pid: 9244]
Object serialized is: Employee Serialization [id=101, name=Soumyajit, dept=IT, designation =Full Stack Developer]
```

Directory Operations

```
1 package Java_AdvanceTraining_Day2_Wipro;
2
3 import java.io.File;
4
5 public class Directory_Operations {
6
7     public static void main(String[] args) {
8         // TODO Auto-generated method stub
9
10        String directoryPath = "C:\\SoumyaPractice.txt";
11
12        //Defining the directory path.
13        File directory = new File (directoryPath);
14
15        //Get all the files from a directory and store in files array.
16
17        File files[] = directory.listFiles();
18
19        //Display all the files in the given directory.
20        System.out.println("List of file in the directory: "+ directoryPath);
21
22
23        if (files!= null) {
24            for (File file: files)
25                //For each loop.. for (<data type> <Object N System.out.println(file.getName());
26                System.out.println(file.getName());
27        }
28
29    }
30 }
31
```

Activate Windows
Go to Settings to activate Windows

The screenshot shows a Java application window titled '<terminated> Directory_Operations [Java Application] C:\Program Files\Java\jdk-21\bin\j'. The window displays the output of the program, which is 'List of file in the directory: C:\\SoumyaPractice.txt'.

Socket Programming

Server Socket

```
1 package Java_AdvanceTraining_Day2_Wipro;
2
3 import java.io.DataInputStream;
4
5 public class Socket_Programming {
6
7     public static void main(String[] args) {
8         // TODO Auto-generated method stub
9
10        try {
11
12            ServerSocket ss = new ServerSocket(6666);
13            System.out.println("Server is waiting at port 6666");
14
15            Socket s = ss.accept(); //Establishes a connection.
16
17            .....  

18
19            //Connected with the client data..
20            DataInputStream dis = new DataInputStream(s.getInputStream());
21            String str = dis.readUTF(); //Reads the client data..
22            System.out.println("Received data from client is: " + str);
23            ss.close();
24        } catch (IOException ex) {
25            System.out.println(ex);
26        }
27    }
28
29
30}
31}
32}
```

Activate Windows

Socket_Programming [Java Application] [pid: 1992]

Server is waiting at port 6666

Client Socket

```
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.Socket;

public class ClientSocket {
    public static void main(String args[]) {
        try {
            //Connecting to server running at localhost at port 6666.
            Socket s = new Socket ("localhost", 6666);

            //Data output stream is connected to Socket
            DataOutputStream dos = new DataOutputStream (s.getOutputStream());

            dos.writeUTF("Hello Java...");

            dos.flush(); //Clears the buffer.

            dos.close();
        } catch (IOException ex) {
            System.out.println(ex);
        }
    }
}

Server is waiting at port 6666
Received data from client is :Hello Java...
```

URL

```
package wipro.Day2;

import java.io.File;
import java.net.MalformedURLException;
import java.net.URL;

public class Test{

    public static void main (String args[]) {

        URL url;
        try {
            url = new URL ("http://www.google.com");
            System.out.println("Protocol : "+url.getProtocol());
            System.out.println("Host name : " + url.getHost());

        } catch (MalformedURLException e) {
            e.printStackTrace();
        }
    }
}
```

INet Address

```
1 package Java_AdvanceTraining_Day2_Wipro;
2
34 import java.net.InetAddress;
4
5
6 public class INetAddress {
7     public static void main (String args[]) {
8
9         try {
10
11             InetAddress IA = InetAddress.getByName ("www.google.com");
12             System.out.println("Host Name : " + IA.getHostName());
13             System.out.println("IP Address: " + IA.getHostAddress());
14
15         } catch (UnknownHostException e) {
16             e.printStackTrace();
17         }
18
19     }
20
21 }
22
```

The screenshot shows an IDE interface with several tabs at the top: JPA Details, Console (which is selected), and Search. Below the tabs is a toolbar with various icons. The main area displays the output of a Java application named 'INetAddress'. The output window title is '<terminated> INetAddress [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe'. The console output shows two lines of text: 'Host Name : www.google.com' and 'IP Address: 142.250.67.68'.

Local Date , Time

```
1 package Java_AdvanceTraining_Day2_Wipro;
2
34 import java.time.LocalDate;
4
5
6 public class LocalDateTime {
7
8     public static void main(String[] args) {
9         // TODO Auto-generated method stub
10        LocalDate ld = LocalDate.now();
11
12        System.out.println("Todays date is: "+ld);
13
14        int day = ld.getDayOfMonth();
15
16        int month = ld.getMonthValue();
17
18        int year = ld.getYear();
19
20        System.out.println("Today's date is: "+day+ "-" + month + "-" +year);
21
22        LocalTime lt = LocalTime.now();
23
24        System.out.println("Current time is: " + lt);
25 }
```

```

25     int hour = lt.getHour();
26
27     int min = lt.getMinute();
28
29     int second = lt.getSecond();
30
31     System.out.println("Current time is : " + hour + ":" + min + ":" + second);
32
33
34
35
36 }
37
38
39 }

```

Activate Windows

The screenshot shows a Java application window with the title bar 'LocalDateTime [Java Application]'. The window contains a toolbar with various icons and tabs labeled 'JPA Details', 'Console X', and 'Search'. The main area displays the following console output:

```

<terminated> LocalDateTime [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe
Todays date is: 1.0
Today's date is: 8-5-2024
Current time is: 16:31:42.931644200
Current time is : 16:31:42

```

TimeZones and ZoneIDs

```

3o import java.time.LocalTime;
4 import java.time.ZoneId;
5 import java.util.Set;
6
7 public class Test{
8
9     public static void main (String args[]) {
0
1         LocalTime lt = LocalTime.now(ZoneId.of("Singapore"));
2         System.out.println("Current time in Singapore : "+lt);
3
4         Set<String> zoneSet = ZoneId.getAvailableZoneIds();
5         System.out.println("List of Zones are : ");
6
7         for (String zone : zoneSet)
8             System.out.println(zone);
9     }
0
1 }

```

List of Zones are :

Asia/Aden
America/Cuiaba
Etc/GMT+9
Etc/GMT+8
Africa/Nairobi
America/Marigot
Asia/Aqtau

DAY 3

Driver Class

Insert-Update-Delete

```
1 package Java_AdvanceTraining_Day3_Wipro;
2
3*import java.sql.Connection;*
4
5 public class Driver_Class {
6
7     public static void main(String[] args) {
8         // TODO Auto-generated method stub
9         try {
10             // Load the driver
11             Class.forName("com.mysql.cj.jdbc.Driver");
12
13             //Connect with the database
14             Connection con = DriverManager.getConnection
15                 ("jdbc:mysql://127.0.0.1:3306/test", "root", "admin" );
16             System.out.println("Connected to the database....");
17
18             Statement st = con.createStatement();
19             st.executeUpdate("INSERT INTO Employee(name, Employee_ID) VALUES('Soumyajit',40000)");
20             con.close();
21         }catch(ClassNotFoundException e) {
22
23             }catch (SQLException ex) {
24                 System.out.println(ex);
25             }
26
27     }
28
29 }
30
31 }
```

Activate Windows

```
<terminated> Driver_Class [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe
Connected to the database....
```

Prepared Statement

```
1 package Java_AdvanceTraining_Day3_Wipro;
2
3 import java.sql.Connection;
4
5 public class Prepared_Statement {
6
7     private static String strQry;
8
9     public static void main(String[] args) {
10         // TODO Auto-generated method stub
11
12         try {
13             //Load the driver.
14             Class.forName ("com.mysql.cj.jdbc.Driver");
15
16             //Connect with the database.
17             Connection con = DriverManager.getConnection
18                 ("jdbc:mysql://127.0.0.1:3306/test", "root", "admin");
19             System.out.println("Connected to the database...");
20
21             String strQry = "INSERT INTO emp (name, sal) VALUES (?,?)";
22             PreparedStatement ps = con.prepareStatement(strQry);
23             Scanner sc = new Scanner (System.in);
24             System.out.println("Enter a name: ");
25             String strName = sc.nextLine();
26             System.out.println("Enter Salary: ");
27             int sal = sc.nextInt();
28
29             int sal = sc.nextInt();
30
31             ps.setString(1, strName);
32
33             ps.setInt(2, sal);
34
35             ps.executeUpdate();
36
37             System.out.println("Inserted Record in the db Successfully..");
38
39             con.close();
40
41             System.out.println("Closed the database connection successfully...");
42
43         }catch (ClassNotFoundException e) {
44
45             } catch (SQLException ex) {
46                 System.out.println(ex);
47             }
48
49         }
50
51     }
52
53 }
54
55
56
```

Activate Windows

Activate Windows

```
Connected to the database...
Enter a name :
Kavya
Enter Salary :
35000
Inserted Record in the db Successfully..
Closed the database connection successfully..
```

ResultSet

```
1 package Java_AdvanceTraining_Day3_Wipro;
2
3 import java.sql.Connection;
4
5 public class ResultSet {
6
7     public static void main(String[] args) {
8         // TODO Auto-generated method stub
9
10     try {
11         //Load the driver.
12         Class.forName ("com.mysql.cj.jdbc.Driver");
13         //Connect with the database.
14
15         Connection con = DriverManager.getConnection
16             ("jdbc:mysql://127.0.0.1:3306/test", "root", "admin");
17             System.out.println("Connected to the database...");
18
19         Statement st = con.createStatement();
20
21         java.sql.ResultSet rs = st.executeQuery("SELECT * FROM emp");
22
23         System.out.println("List of records in emp table are: ");
24
25         while (rs.next()) {
26
27             System.out.print(rs.getString(1));
28             System.out.println(" " + rs.getInt(2));
29             con.close();
30             System.out.println("Closed the database connection successfully...");
31         }
32         catch (ClassNotFoundException e) {
33
34     } catch (SQLException ex) {
35
36     System.out.println(ex);
37 }
38
39 }
40
41
42 }
43
44     public static String getInt(String string) {
45         // TODO Auto-generated method stub
46         return null;
47     }
48
49     public static String getString(String string) {
50         // TODO Auto-generated method stub
51         return null;
52     }
53
54     public static boolean next() {
55         // TODO Auto-generated method stub
56         return false;
57     }
58 }
```

```
Connected to the database...
List of records in emp table are :
Charan.K 1000
Shiva 1000
Sai Rakesh 30000
Trupti 30000
Kavya 35000
Closed the database connection successfully...
```

Activate Windows

Activate Windows

JDBC MenuApp

```
1 package Java_AdvanceTraining_Day3_Wipro;
2
3 import java.sql.Connection;
4
5 public class JDBCMenuApp {
6
7     // Database connection details
8     private static final String DB_URL = "jdbc:mysql://127.0.0.1:3306/test"; // Modify as needed
9     private static final String DB_USER = "root"; // Modify as needed
10    private static final String DB_PASSWORD = "admin"; // Modify as needed
11
12    public static void main(String[] args) {
13        // TODO Auto-generated method stub
14
15        try (Connection conn = DriverManager.getConnection(DB_URL, DB_USER, DB_PASSWORD)) {
16            Scanner scanner = new Scanner(System.in);
17            boolean exit = false;
18            while (!exit) {
19                System.out.println("Menu:");
20                System.out.println("1. Insert");
21                System.out.println("2. Update");
22                System.out.println("3. Delete");
23                System.out.println("4. Display data (Select)");
24                System.out.println("5. Exit");
25                System.out.print("Enter your choice: (1-5 )");
26
27                int choice = scanner.nextInt();
28                scanner.nextLine();
29                switch (choice) {
30                    case 1:
31                        insertRecord(conn, scanner);
32                        break;
33                    case 2:
34                        updateRecord(conn, scanner);
35                        break;
36                    case 3:
37                        deleteRecord(conn, scanner);
38                        break;
39                    case 4:
40                        displayRecords(conn);
41                        break;
42                    case 5:
43                        exit = true;
44                        System.out.println("Exiting...");
45                        break;
46                    default:
47                        System.out.println("Invalid choice, please try again.");
48                        break;
49                }
50            }
51        }
52        scanner.close();
53    }
54}
```

```
60 } catch (SQLException e) {
61     System.err.println("Database connection error: " + e.getMessage());
62 }
63 }
64 private static void insertRecord(Connection conn, Scanner scanner) throws SQLException {
65     System.out.print("Enter name: ");
66     String name = scanner.nextLine();
67
68     System.out.print("Enter age: ");
69     int age = scanner.nextInt();
70
71     String sql = "INSERT INTO your_table (name, age) VALUES (?, ?)";
72     try (PreparedStatement pstmt = conn.prepareStatement(sql)) {
73         pstmt.setString(1, name);
74         pstmt.setInt(2, age);
75         pstmt.executeUpdate();
76         System.out.println("Record inserted successfully.");
77     }
78 }
79
80 private static void updateRecord(Connection conn, Scanner scanner) throws SQLException {
81     System.out.print("Enter record ID to update: ");
82     int id = scanner.nextInt();
83     scanner.nextLine(); // Consume newline
84
85     System.out.print("Enter new name: ");
86     String newName = scanner.nextLine();
87
88     System.out.print("Enter new age: ");
89     int newAge = scanner.nextInt();
90
91     String sql = "UPDATE your_table SET name = ?, age = ? WHERE id = ?";
92     try (PreparedStatement pstmt = conn.prepareStatement(sql)) {
93         pstmt.setString(1, newName);
94         pstmt.setInt(2, newAge);
95         pstmt.setInt(3, id);
96         pstmt.executeUpdate();
97         System.out.println("Record updated successfully.");
98     }
99 }
100
101 private static void deleteRecord(Connection conn, Scanner scanner) throws SQLException {
102     System.out.print("Enter record ID to delete: ");
103     int id = scanner.nextInt();
104
105     String sql = "DELETE FROM your_table WHERE id = ?";
106     try (PreparedStatement pstmt = conn.prepareStatement(sql)) {
107         pstmt.setInt(1, id);
108         pstmt.executeUpdate();
109         System.out.println("Record deleted successfully.");
110     }
111 }
112
113 private static void displayRecords(Connection conn) throws SQLException {
114     String sql = "SELECT * FROM your_table"; // Replace 'your_table' with your actual table name
115     try (Statement stmt = conn.createStatement();
116          ResultSet rs = stmt.executeQuery(sql)) {
117         System.out.println("ID | Name | Age");
118         while (rs.next()) {
119             System.out.println(rs.getInt("id") + " | " + rs.getString("name") + " | " +
120             rs.getInt("age"));
121         }
122     }
123 }
124 }
125 }
```

Activate Windows

Activate Windows

```
<terminated> JDBCMenuApp [Java Application] C:\Program Files\Java\jdk-21\bin\java
Menu:
1. Insert
2. Update
3. Delete
4. Display data (Select)
5. Exit
Enter your choice: (1-5)
1
Enter name: Soumya
Enter age: 25
```

DataBase MetaData

```
1 package Java_AdvanceTraining_Day3_Wipro;
2
3@import java.sql.SQLException;I
4
5 public class Database_MetaData {
6
7     public static void main(String[] args) {
8         // TODO Auto-generated method stub
9
10    try {
11
12        //Load the driver.
13
14        Class.forName ("com.mysql.cj.jdbc.Driver");I
15
16        //Connect with the database.
17
18        Connection con = DriverManager.getConnection("jdbc:mysql://127.0.0.1:3306/test",
19            "root", "admin");
20        System.out.println("Connected to the database...");I
21
22        DatabaseMetaData dbMeta = con.getMetaData();
23
24
25        System.out.println("Name of the database: "+dbMeta.getDatabaseProductName());
26        System.out.println("Name of the driver: "+dbMeta.getDriverName());I
27
28
29
30 }
```

```
31 System.out.println("Driver version:" + dbMeta.getDriverVersion());
32
33 System.out.println("Table length name: " +dbMeta.getMaxTableNameLength());
34
35 System.out.println("Maximum number of columns in a table: " +dbMeta.getMaxColumnsInTable());
36
37
38 con.close();
39
40 System.out.println("Closed the database connection successfully...");
41
42 System.out.println("Closed the database connection successfully...");
43 } catch (ClassNotFoundException e) {
44
45 } catch (SQLException ex) {
46
47 System.out.println(ex);
48
49 }
50
51
52
53 }
```

<terminated> Database_MetaData [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (08-May-2024, 5:13:24 pm – 5:11)
Connected to the database...
Name of the database: MySQL
Name of the driver: MySQL Connector/J
Driver version:mysql-connector-j-8.3.0 (Revision: 805f872a57875f311cb82487efcfb070411a3fa0)
Table length name: 64
Maximum number of columns in a table: 512
Closed the database connection successfully...
Closed the database connection successfully...

Junit

```
1 package Java_AdvanceTraining_Day3_Wipro;
2
3 import org.junit.Test;
4
5 public class Junit_Suites {
6     public int Junit_Suites(int x, int y) {
7         return x * y;
8     }
9
10    public void Junit_Suites() {
11        Junit_Suites testObj = new Junit_Suites();
12        int x = 20, y = 3, finalResult = 60;
13
14        int functionResult = testObj.Junit_Suites(x, y);
15        assertEquals (finalResult, functionResult);
16    }
17
18    private void assertEquals(int finalResult, int functionResult) {
19        // TODO Auto-generated method stub
20
21    }
22
23
24 }
```

Runs: 1/1 Errors: 0 Failures: 0

DAY 4

Design Pattern

Singleton Pattern

```
1 package Java_AdvanceTraining_Day4_Wipro;
2
3 class Whatsapp{
4     private static Whatsapp whatsappObject = null;
5     private Whatsapp() {}
6     public static Whatsapp getInstance() {
7         if(whatsappObject == null)
8             whatsappObject = new Whatsapp();
9         return whatsappObject;
10    }
11 }
12
13 public class Singleton {
14
15     public static void main(String args []) {
16         // TODO Auto-generated method stub
17
18         Whatsapp whatsappObjFirstTime = Whatsapp.getInstance();
19         System.out.println("Memory location of the object :"+
20                             whatsappObjFirstTime);
21
22     }
23
24 }
```

```
<terminated> Singleton [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (08-May-2024, 5:20:58
Memory location of the object :Java_AdvanceTraining_Day4_Wipro.Whatsapp@5e9f23b4
```

Factory Pattern

```
1 package Java_AdvanceTraining_Day4_Wipro;
2
3 import java.util.Scanner;
4
5 abstract class Plan1{
6     double ratePerUnit;
7     abstract void getRate();
8     public void calculateBill (int units) {
9         System.out.println("Consumer has to pay Rs." +units+ratePerUnit);
10    }
11 }
12
13
14 abstract class DomesticPlan1 extends Plan1{
15     public void getRate() {
16         ratePerUnit = 3.50;
17     }
18 }
19
20 class InstitutionPlan1 extends Plan1{
21     public void getRate() {
22         ratePerUnit = 5.50;
23     }
24 }
25
26 class CommercialPlan1 extends Plan1{
27     public void getRate() {
28         ratePerUnit = 7.50;
29     }
30
31
32 class GetPlanFactory1{
33     public Plan1 getPlan(String planType) {
34         if(planType.equals("Domestic"))
35             return new InstitutionPlan1();
36         if(planType.equals("Institution"))
37             return new InstitutionPlan1();
38         else if(planType.equals("Commercial"))
39             return new CommercialPlan1();
40         return null;
41     }
42 }
43
44 public class FactoryPattern {
45
46     public static void main(String[] args) {
47         // TODO Auto-generated method stub
48
49         Scanner sc = new Scanner(System.in);
50         System.out.println
51 ("Enter the type of building : (Domestic/Institution/Commercial)");
52         String strBuildingType = sc.next();
53
54         GetPlanFactory1 planFactory = new GetPlanFactory1();
55         Plan1 pln = planFactory.getPlan(strBuildingType);
56 }
```

Activate Windows

Activate Windows

```

56
57     System.out.println("How many units of consumed...");
58     int iUnits = sc.nextInt();
59     pln.getRate();
60
61     System.out.println("Rate per Unit for"+ strBuildingType
62                     +"is Rs.:"+pln.ratePerUnit);
63     pln.calculateBill(iUnits);
64
65 }
66
67 }

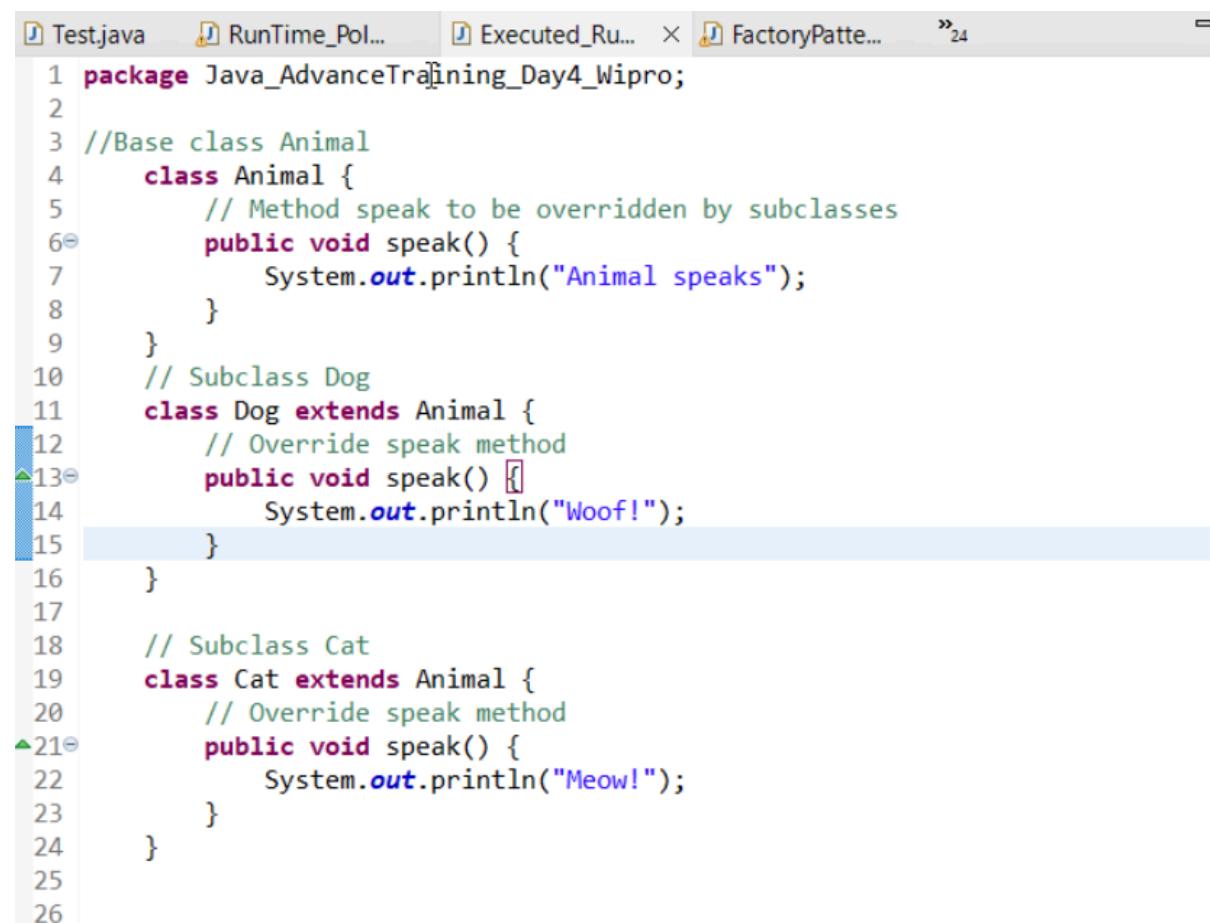
```

```

<terminated> FactoryPattern [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe
Enter the type of building : (Domestic/Institution/Commercial) 
Institution
How many units of consumed...
1
Rate per Unit for Institution is Rs.:5.5
Consumer has to pay Rs.15.5

```

Executed RunTime Polymorphism



```

1 package Java_AdvanceTraining_Day4_Wipro;
2
3 //Base class Animal
4 class Animal {
5     // Method speak to be overridden by subclasses
6     public void speak() {
7         System.out.println("Animal speaks");
8     }
9 }
10 // Subclass Dog
11 class Dog extends Animal {
12     // Override speak method
13     public void speak() {
14         System.out.println("Woof!");
15     }
16 }
17
18 // Subclass Cat
19 class Cat extends Animal {
20     // Override speak method
21     public void speak() {
22         System.out.println("Meow!");
23     }
24 }
25
26

```

```
27  
28 public class Executed_RunTime_Polymorphism {  
29  
30     public static void makeAnimalSpeak(Animal animal) {  
31         animal.speak();  
32     }  
33  
34     public static void main(String[] args) {  
35         // TODO Auto-generated method stub  
36  
37         Animal Dog = new Dog();  
38         Animal Cat = new Cat();  
39  
40         makeAnimalSpeak(Dog);  
41         makeAnimalSpeak(Cat);  
42  
43     }  
44  
45 }  
46  
47 }
```

The screenshot shows an IDE interface with the following details:

- Toolbar:** Task List, Problems, Javadoc, Declaration, Console (selected), Coverage.
- Console Tab:** Shows the output of the application execution.
- Output Content:**

```
<terminated> Executed_RunTime_Polymorphism [Java Application] C:\Program Files\Java\  
Woof!  
Meow!
```

RunTime Polymorphism

The screenshot shows an IDE interface with a code editor and a console window.

Code Editor:

```
1 package Java_AdvanceTraining_Day4_Wipro;
2
3 class Test{
4     public void method() {
5         System.out.println("Method1");
6     }
7 }
8
9
10 public class RunTime_Polymorphism{
11
12     private static final Test test = null;
13
14     public static void main(String[] args) {
15         // TODO Auto-generated method stub
16
17         Test test = new Test();
18         test.method();
19
20     }
21
22 }
23
```

Console Window:

```
<terminated> RunTime_Polymorphism [Java Application] C:\Program Files\Java\jdk-21\bin
Method1
```

DAY 5

Builder Pattern

```
1 package Java_AdvanceTraining_Day5_Wipro;
2
3 class ShoppingItem {
4     private String itemName;
5     int quantity;
6     private double pricePerUnit;
7
8     public ShoppingItem(String itemName, int quantity,
9             double pricePerUnit) {
10         this.itemName = itemName;
11         this.quantity = quantity;
12         this.pricePerUnit = pricePerUnit;
13     }
14
15     public String getItemName() {
16         return itemName;
17     }
18
19     public int getQuantity() {
20         return quantity;
21     }
22
23     public double getPricePerUnit() {
24         return pricePerUnit;
25     }
26
27     public double getTotalPrice() {
28         return quantity * pricePerUnit;
29     }
30
31     @Override
32     public String toString() {
33         return "Item: " + itemName + ", Quantity: " + quantity +
34             ", Price per unit: $" + pricePerUnit +
35             ", Total Price: $" + getTotalPrice();
36     }
37 }
38
39 class ShoppingBuilder {
40     private ShoppingItem item;
41
42     public ShoppingBuilder(String itemName, int quantity,
43             double pricePerUnit) {
44         this.item = new ShoppingItem(itemName, quantity,
45             pricePerUnit);
46     }
47
48     public ShoppingBuilder setQuantity(int quantity) {
49         this.item.quantity = quantity;
50         return this;
51     }
52 }
```

Activate Windows

```

53     public ShoppingItem build() {
54         return this.item;
55     }
56 }
57
58 public class BuilderPattern {
59     public static void main(String[] args) {
60         ShoppingItem groceryItem = new ShoppingBuilder
61             ("Grocery", 2, 5.0).build();
62         System.out.println(groceryItem);
63
64         ShoppingItem vegetableItem = new ShoppingBuilder
65             ("Vegetable", 1, 3.5)
66             .setQuantity(3)
67             .build();
68         System.out.println(vegetableItem);
69
70         ShoppingItem fruitItem = new ShoppingBuilder("Fruit",
71             5, 2.0)
72             .setQuantity(10)
73             .build();
74         System.out.println(fruitItem);
75     }
76 }

```

Activate Windows

```

<terminated> BuilderPattern [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (08-
Item: Grocery, Quantity: 2, Price per unit: $5.0, Total Price: $10.0
Item: Vegetable, Quantity: 3, Price per unit: $3.5, Total Price: $10.5
Item: Fruit, Quantity: 10, Price per unit: $2.0, Total Price: $20.0

```

Bank Loan

```

1 package Java_AdvanceTraining_Day5_Wipro;
2
3 interface Loan {
4     double calculateInterest();
5 }
6
7 class EducationalLoan implements Loan {
8     @Override
9     public double calculateInterest() {
10         return 0.10;
11     }
12 }
13
14 class HousingLoan implements Loan {
15     @Override
16     public double calculateInterest() {
17         return 0.12;
18     }
19 }
20
21 class BusinessLoan implements Loan {
22     @Override
23     public double calculateInterest() {
24         return 0.15;
25     }
26 }

```

Activate Windows

```
28 class LoanFactory {
29     public static Loan getLoan(String loanType) {
30         switch (loanType.toLowerCase()) {
31             case "educational":
32                 return new EducationalLoan();
33             case "housing":
34                 return new HousingLoan();
35             case "business":
36                 return new BusinessLoan();
37             default:
38                 return null;
39         }
40     }
41 }
42
43 public class Bank_Loan {
44     public static void main(String[] args) {
45         String loanType = "business"; // You can change this to any type of loan
46         Loan loan = LoanFactory.getLoan(loanType);
47         if (loan != null) {
48             double interestRate = loan.calculateInterest();
49             System.out.println("The interest rate for " +
50                 loanType + " loan is " + (interestRate * 100) + "%");
51         } else {
52             System.out.println("Invalid loan type");
53         }
54     }
55 }
```

Activate Windows



```
<terminated> Bank_Loan [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\java.exe -jar C:\Users\Public\Documents\NetBeansProjects\Bank_Loan\dist\Bank_Loan.jar
The interest rate for business loan is 15.0%
```

DAY 6

Adapter Pattern

```
1 package Java_AdvanceTraining_Day6_Wipro;
2
3 interface MediaPlayer1{
4
5     public void play (String audioType, String fileName); }
6
7 interface AdvancedMediaPlayer1{
8     public void playVLC (String fileName);
9     public void playMp4 (String fileName);
10    }
11
12 //Step: 2
13 class VLCPlayer1 implements AdvancedMediaPlayer1{
14 @Override
15 public void playVLC (String fileName) {
16     System.out.println("Playing VLC file name: "+fileName);
17
18 }                                I
19 @Override
20 public void playMp4 (String fileName) {
21 //do nothing.
22 }
23 }
24
25 class MP4Player1 implements AdvancedMediaPlayer1 {
26 @Override
27 public void playVLC (String fileName) {
28 //do nothing..
29
30 }
31
32 @Override
33 public void playMp4 (String fileName) {
34 System.out.println("Playing MP4 for file name: "+fileName);
35
36 }
37 }
38
39 //Step: 3
40
41 class MediaAdapter1 implements MediaPlayer1{
42 AdvancedMediaPlayer1 advancedMediaPlayer;
43 @Override
44 public MediaAdapter1 (String audioType) {
45     if (audioType.equalsIgnoreCase ("vlc"))
46         advancedMediaPlayer = new VLCPlayer1();
47
48     else if (audioType.equalsIgnoreCase("mp4"))
49         advancedMediaPlayer = new MP4Player1();
50 }
51
52 @Override
53 public void play (String audioType, String fileName) {
54     if (audioType.equalsIgnoreCase ("mp4"))
55         advancedMediaPlayer.playMp4 (fileName);
56     else if (audioType.equalsIgnoreCase("vlc"))
```

Activate Windows

Activate Windows

```

57     advanedMediaPlayer.playVLC(fileName);
58 }
59 }
60 }
61
62 //Step: 4
63 class AudioPlayerl implements MediaPlayerl {
64     MediaAdapterl mediaAdapter;
65 @Override
66 public void play (String audioType, String fileName) {
67 if (audioType.equalsIgnoreCase("mp3"))
68 System.out.println("Playing Mp3 for the file name "+fileName);
69 else if (audioType.equalsIgnoreCase("vlc") || audioType.equalsIgnoreCase("mp4")) {
70     mediaAdapter = new MediaAdapterl (audioType); //Runtime polymorphism is created..
71     mediaAdapter.play(audioType, fileName);
72 } //Corresponding file is played..
73
74 else
75 System.out.println("Invalid Media...");
76 }
77 }
78
79 public class Adapter_Pattern {
80
81@     public static void main(String[] args) {
82         // TODO Auto-generated method stub
83
84         AudioPlayerl audioPlayer = new AudioPlayerl();
85         audioPlayer.play("mp3", "123 file..");
86         audioPlayer.play("mp4", "23456 file..");
87         audioPlayer.play("vlc", "vlc file..");
88         audioPlayer.play("avi", "avi file");
89
90     }
91
92 }
93

```

Activate Windows

```

<terminated> Adapter_Pattern (1) [Java Application] C:\Program Files\Java\jdk-21\
Playing Mp3 for the file name 123 file..
Playing MP4 for file name: 23456 file..
Playing VLC file name: vlc file..
Invalid Media...

```

JSP and Servlet

Scriptlet

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4<html>
5<head>
6 <meta charset="UTF-8">
7 <title>Insert title here</title>
8 </head>
9<body>
10 <h1>I am Soumyajit. I am from Burdwan, West Bengal </h1>
11 |
12<%
13 int a = 220, b = 380, c;
14 c = a + b;
15 out.println("Result is :" +c);
16 %>
17
18
19
20
21 </body>
22 </html>
```

← → ⌂ ⓘ localhost:8085/Test_JSP/NewFile.jsp

I am Soumyajit. I am from Burdwan, West Bengal

Declarative Tags

```
1  %@ page language="java" contentType="text/html; charset=UTF-8"
2      pageEncoding="UTF-8"%>
3  <!DOCTYPE html>
4  <html>
5  <head>
6  <meta charset="UTF-8">
7  <title>Insert title here</title>
8  </head>
9  <body>
10 <h1> Factorial using Declarative and Scriptlet tags </h1>
11
12<%!
13 int factorial(int n){
14     int fact =1;
15     for(int i =1; i <=n; ++i) {
16         fact = fact * i;
17     return fact;
18 }
19 %>
20<%
21 int result = factorial(5);
22 out.println("Factorial of 5 is : "+ result);
23 %>
24
25</body>
26</html>
```

Activate Windows

← → ⌂ localhost:8085/Test_JSP/Factorial.jsp

Factorial using Declarative and Scriptlet tags

Factorial of 5 is : 120

Prime or Not

```
1  %@ page language="java" contentType="text/html; charset=UTF-8"
2      pageEncoding="UTF-8"%>
3  <!DOCTYPE html>
4  <html>
5  <head>
6  <meta charset="UTF-8">
7  <title>Prime or not</title>
8  </head>
9  <body>
10 <h1>Prime Number Checker</h1>
11 <form method="post">
12     Enter a number: <input type="text" name="number">
13     <input type="submit" value="Check">
14 </form>
15
16<%
17     String numberStr = request.getParameter("number");
18     if (numberStr != null && !numberStr.isEmpty()) {
19         int number = Integer.parseInt(numberStr);
20         boolean isPrime = true;
```

```
22     if (number <= 1) {
23         isPrime = false;
24     } else {
25         for (int i = 2; i <= Math.sqrt(number); i++) {
26             if (number % i == 0) {
27                 isPrime = false;
28                 break;
29             }
30         }
31     }
32 %>
33
34 <p>The number <%= number %> is <%= isPrime ? "prime" : "not prime" %>.</p>
35
36 <% } %>
37
38 </body>
39 </html>
```

Activate Windows

← → ⌂ ⓘ localhost:8085/Test_JSP/Prime%20or%20Not.jsp

Prime Number Checker

Enter a number: Check

The number 6 is not prime.

← → ⌂ ⓘ localhost:8085/Test_JSP/Prime%20or%20Not.jsp

Prime Number Checker

Enter a number: Check

The number 2 is prime.

Spring Core

Displaying the bean info

Pom.xml

```
https://maven.apache.org/xsd/maven-4.0.0.xsd (xsi:schemaLocation)
3<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4  <modelVersion>4.0.0</modelVersion>
5  <groupId>EricsonTest</groupId>
6  <artifactId>EricsonTest</artifactId>
7  <version>0.0.1-SNAPSHOT</version>
8  <name>EricsonTest</name>
9  <description>EricsonTest</description>
10 <dependencies>
11   <dependency>
12     <groupId>org.springframework</groupId>
13     <artifactId>spring-core</artifactId>
14     <version>5.2.16.RELEASE</version>
15   </dependency>
16   <dependency>
17     <groupId>org.springframework</groupId>
18     <artifactId>spring-context</artifactId>
19   </dependency>
20 </dependencies>
21 </project>
22
23
```

Employee1.java

```
1 |
2 public class Employee1 {
3
4     int id;
5     String name;
6     String dept;
7     String designation;
8
9     Employee1(){}
10    public Employee1(int id, String name, String dept, String designation) {
11        this.id = id;
12        this.name = name;
13        this.dept = dept;
14        this.designation = designation;
15    }
16
17
18
19
20    public int getId() {
21        return id;
22    }
23
24    public void setId(int id) {
25        this.id = id;
26    }
27
```

Activate Windows

```

28*     public String getName() {
29         return name;
30     }
31
32*     public void setName(String name) {
33         this.name = name;
34     }
35
36*     public String getDept() {
37         return dept;
38     }
39
40*     public void setDept(String dept) {
41         this.dept = dept;
42     }
43
44*     public String getDesignation() {
45         return designation;
46     }
47
48*     public void setDesignation(String designation) {
49         this.designation = designation;
50     }
51
52 }

```

Activate Windows

display_Bean_Info.java

```

1+ import org.springframework.context.ApplicationContext;□
2
3
4
5 public class display_Bean_Info {
6
7     public static void main(String[] args) {
8         // TODO Auto-generated method stub
9         ApplicationContext context = new ClassPathXmlApplicationContext("display_BeanInfo_Hellobean.xml");
10
11         Employee1 empObj = (Employee1) context.getBean("emp");
12
13         System.out.println("Employee details...");
14
15         System.out.println("Id: "+empObj.getId());
16
17         System.out.println("Name: "+empObj.getName());
18         System.out.println("Dept: "+empObj.getDept());
19
20         System.out.println("Designation: "+empObj.getDesignation());
21
22     }
23
24
25

```

```
<terminated> display_Bean_Info [Java Application] C:\Program Files\Java\jdk-21\bin  
Employee details...  
Id: 102  
Name: Soumyajit Mondal  
Dept: Technology  
Designation: Full Stack Developer
```

Spring MVC

My Controller

MyController.java

```
1 package com.soumya.demo.controller;  
2  
3 import org.springframework.stereotype.Controller;  
4  
5 @Controller  
6 public class MyController {  
7  
8     // when the user gives in url as http://localhost:8089/  
9     // inside the controller , it will be looked for "/" and the function  
10    // which below it is executed automatically  
11    @GetMapping("/")  
12    public String displayGreeting() {  
13        return "welcome";  
14    }  
15  
16    //http://localhost:8089/login  
17    @GetMapping("/login")  
18    public String displayLogin() {  
19        return "login"; //WEB-INF\jsp\login.jsp  
20    }  
21  
22    @PostMapping("/login")  
23    public String loginValidation(ModelMap model, @RequestParam String userID,  
24                                    @RequestParam String password) {  
25        System.out.println("Received data : User ID:" + userID + "password :" + password);  
26        String str = "Invalid User";  
27  
28        if(userID.equals("Soumya")&& password.equals("Admin"))  
29            str = "Valid User";  
30  
31        model.put("message", str);  
32        return "results";  
33    }  
34  
35 }  
36  
37 }  
38  
39
```

Application.properties

```
1 spring.application.name=SpringMVC
2 server.port=8089
3 spring.mvc.view.prefix=/WEB-INF/jsp/
4 spring.mvc.view.suffix=.jsp|
```

Welcome.jsp File

```
1<h1> Hii Sir , Myself Soumyajit. </h1>
```

← → ⌂ ⚡ localhost:8089

Hii Sir , Myself Soumyajit.

Login Validation

```
1 package com.soumya.demo.controller;
2
3 import org.springframework.stereotype.Controller;
4
5 @Controller
6 public class MyController {
7
8     // when the user gives in url as http://localhost:8089/
9     // inside the controller , it will be looked for "/" and the function
10    // which below it is executed automatically
11    @GetMapping("/")
12    public String displayGreeting() {
13        return "welcome";
14    }
15
16    //http://localhost:8089/login
17    @GetMapping("/login")
18    public String displayLogin() {
19        return "login"; //WEB-INF\jsp\login.jsp
20    }
21
22    @PostMapping("/login")
23    public String loginValidation(ModelMap model , @RequestParam String userID,
24                                @RequestParam String password) {
25        System.out.println("Received data : User ID:" + userID + "password :" + password);
26        String str = "Invalid User";
27
28        if(userID.equals("Soumya")&& password.equals("Admin"))
29            str = "Valid User";
30
31        model.put("message", str);
32        return "results";
33    }
34
35 }
```

Login.jsp File

```
1<html>
2<body>
3
4<h1> Login Form </h1>
5<FORM method = "post">
6User ID : <Input type = "text" name ="userID"/> <br/> <br/>
7Password : <Input type = "text" name ="password"/> <br/> <br/>
8<Input type = "Submit"/>
9
10</FORM>
11</body>
12</html>
```

Results.jsp File

```
1<h1>${message}</h1>
```

Run the program in the browser

<http://localhost:8089/login>

← → ⌂ localhost:8089/login

Login Form

User ID :

Password :

← → ⌂ localhost:8089/login

Login Form

User ID :

Password :

← → ⌂ ⓘ localhost:8089/login

Valid User

Spring REST

Application.properties

```
application.properties ×  
1 spring.application.name=SpringREST  
2 server.port=8091  
3
```

Welcome

← → ⌂ ⓘ localhost:8091/welcome



Welcome to Spring REST from Soumya...

Login Validation

```
1 package com.soumya.demo.controller;
2
3 import org.springframework.web.bind.annotation.GetMapping;
4
5 @RestController
6 public class MyController {
7
8     @GetMapping ("/welcome")
9     public String welcomeMessage() {
10         String str = "Welcome to Spring REST from Soumya...";
11         return str;
12     }
13
14     /**
15      * Read the parameter from the client.
16      */
17     @GetMapping ("/login")
18     public String loginValidation(@RequestParam String UID, @RequestParam String pwd) {
19         System.out.println("Given credentials are : Login" + UID+ "Password:" +pwd);
20         if (UID.equals("Soumya")&& pwd.equals("Admin"))
21             return "You are a valid user";
22         else
23             return "Invalid user !! Check your credentials";
24     }
25 }
```

Output



A screenshot of a web browser window. The address bar shows the URL `localhost:8091/login?UID=Soumya&pwd=Admin`. The page content displays the message `You are a valid user`.



A screenshot of a web browser window. The address bar shows the URL `localhost:8091/login?UID=Soumya&pwd=Admin22`. The page content displays the message `Invalid user !! Check your credentials`.

POJO Class - Employee.java

```
1 package com.soumya.demo.data;
2
3 public class Employee {
4
5     int id;
6     String name;
7     String dept;
8     String designation;
9
10
11    public Employee(int id, String name, String dept, String designation) {
12        super();
13        this.id = id;
14        this.name = name;
15        this.dept = dept;
16        this.designation = designation;
17    }
18
19    public int getId() {
20        return id;
21    }
22
23    public void setId(int id) {
24        this.id = id;
25    }
26
27    public String getName() {
28        return name;
29    }
30
31    public void setName(String name) {
32        this.name = name;
33    }
34
35    public String getDept() {
36        return dept;
37    }
38
39    public void setDept(String dept) {
40        this.dept = dept;
41    }
42
43    public String getDesignation() {
44        return designation;
45    }
46
47    public void setDesignation(String designation) {
48        this.designation = designation;
49    }
50
51
52    @Override
53    public String toString() {
54        return "Employee [id=" + id + ", name=" + name + ", dept=" + dept + ", designation=" + designation + "]";
55    }
56
57
58 }
```

Activate Windows

MyController.java

CRUD Operations

```
33
34@PostMapping("/insertData")
35 public String insertData(@RequestBody Employee empObj) {
36     System.out.println("Given data from client is : " + empObj);
37     return "Given data is inserted in the database successfully...";
38 }
39
40@PutMapping("/updateData")
41 public String updateData(@RequestBody Employee empObj) {
42     System.out.println("Given Object to update is : "+empObj);
43     return "Given data is updated in the database successfully...";
44 }
45
46@DeleteMapping("/deleteById/{id}")
47 public String deleteByID(@PathVariable int id) {
48     String strReturn = "Employee with id "+id+"is deleted successfully from the database";
49     return strReturn;
50 }
51
52 }
53 
```

Activate Windows

[Go to Settings to activate Windows](#)

Run the Server

Check POSTMAN

POST Operation

The screenshot shows the POSTMAN interface with the following details:

- Method:** POST
- URL:** http://localhost:8091/insertData
- Body (JSON):**

```
1 {
2   "id":2323233,
3   "name":"Soumyajit Mondal",
4   "dept":"IT",
5   "designation":"Full Stack Developer"
```
- Response Status:** 200 OK
- Response Time:** 21 ms
- Response Size:** 204 B
- Response Content:** Record Inserted in Database Successfully

Activate Windows

PUT Operation

The screenshot shows the Postman interface for a PUT operation. The URL is `http://localhost:8091/updateData`. The request method is set to `PUT`. The `Body` tab is selected, showing a JSON payload:

```
1 {  
2   "id":2323233,  
3   "name":"Soumyajit",  
4   "dept":"IT",  
5   "designation":"Full Stack Developer"
```

The response status is `200 OK` with a time of `130 ms` and a size of `217 B`. The response body is:

```
1 Given data is updated in the database successfully...Activate Windows
```

Given Object to update is : Employee [id=2323233, name=Soumyajit, dept=IT, designation=Full Stack Developer] Activate

DELETE Operation

The screenshot shows the Postman interface for a DELETE operation. The URL is `http://localhost:8091/deleteByID/102`. The request method is set to `DELETE`. The `Body` tab is selected, showing a table for Query Params:

	Key	Value	Bulk Edit
	Key	Value	

The response status is `404 Not Found` with a time of `37 ms` and a size of `4.83 KB`. The response body is:

```
1
```

Activate Windows

Spring UNIT Test

MyController

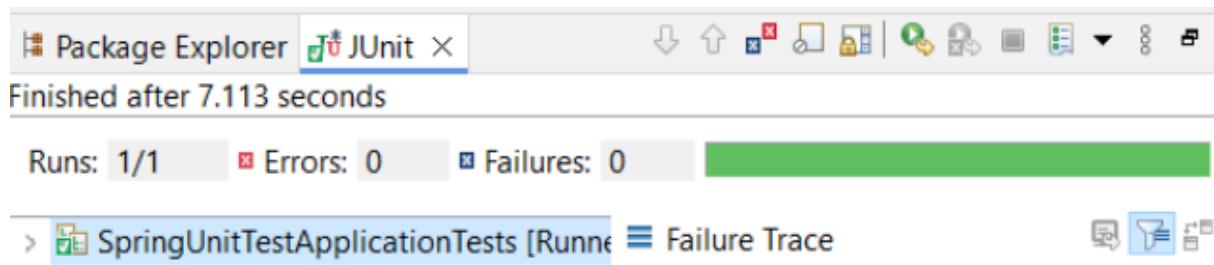
```
1 package com.soumya.demo.controller;
2
3 import org.springframework.web.bind.annotation.GetMapping;
4
5
6 @RestController
7
8 public class MyController {
9
10    @GetMapping("/login")
11    public String login(String UID, String PWD) {
12        if(UID.equals("Soumya")&& PWD.equals("Admin"))
13            return "You are a Valid User";
14        else
15            return "You are a Invalid User";
16    }
17
18 }
19
```

SpringUnitIntegrationTestingTests.java.

SpringUnitTestApplicationTests.java

```
1 package com.soumya.demo;
2
3 import static org.junit.jupiter.api.Assertions.assertEquals;
4
5
6 @SpringBootTest
7 class SpringUnitTestApplicationTests {
8
9     @Autowired
10     MyController myController;
11
12     @Test
13     void contextLoads() {
14
15         String UID="Soumya", PWD="Admin";
16         String strValidUser = "You are a Valid User";
17         String strInvalidUser= "You are an Invalid User";
18
19         String strResult = myController.login(UID, PWD);
20         assertEquals(strValidUser,strResult);
21     }
22
23 }
24
```

Output



Hibernate

CRUD Operation

hibernate.cfg.xml File

```
//Hibernate/Hibernate Configuration DTD 3.0//EN (doctype)
1 |?xml version="1.0" encoding="UTF-8"?>
2 |<!DOCTYPE hibernate-configuration PUBLIC
3 |
4 |"-//Hibernate/Hibernate Configuration DTD 3.0//EN"
5 |"http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
6 |<hibernate-configuration>
7 |<session-factory>
8 |<property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
9 |<property name="hibernate.connection.url">jdbc:mysql://127.0.0.1:3306/test</property>
10 |<property name="hibernate.connection.username">root</property>
11 |<property name="connection.password">admin</property>
12 |<property name="connection.pool_size">10</property>
13 |<property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
14 |<property name="show_sql">true</property>
15 |<property name="hibernate.hbm2ddl.auto">update</property>
16 |<mapping resource="employee.hbm.xml"/>
17 |</session-factory>
18 |</hibernate-configuration>
```

POJO Class

Employee.java

```
1 package HibernateCRUD;
2 public class Employee {
3
4     int id;
5     String firstName;
6     String lastName;
7     int salary;
8
9     Employee(){}
10
11    public Employee(int id, String firstName, String lastName, int salary) {
12        super();
13        this.id = id;
14        this.firstName = firstName;
15        this.lastName = lastName;
16        this.salary = salary;
17    }
18
19    public int getId() {
20        return id;
21    }
22
23    public void setId(int id) {
24        this.id = id;
25    }
26
27    public String getFirstName() {
28        return firstName;
29    }
30
31    public void setFirstName(String firstName) {
32        this.firstName = firstName;
33    }
34
35    public String getLastName() {
36        return lastName;
37    }
38
39    public void setLastName(String lastName) {
40        this.lastName = lastName;
41    }
42
43    public int getSalary() {
44        return salary;
45    }
46
47    public void setSalary(int salary) {
48        this.salary = salary;
49    }
50
51    |
52 }
```

Activate Windows

employee.hbm.xml File

```
-//Hibernate/Hibernate Mapping DTD//EN (doctype)
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE hibernate-mapping PUBLIC
3
4 "-//Hibernate/Hibernate Mapping DTD//EN"
5 "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
6<hibernate-mapping>
7<class name="Employee" table="hibernate_create_insert_update_delete">
8 <id name="id" type="int" column="id"> </id>
9 <property name="firstName" column="first_name" type="string"/>
10 <Element type "property" must be declared. "last_name" type="string"/>
11 <property name="salary" type="int"/>
12
13 </class>
14
15 </hibernate-mapping>
```

Client.java

```
import java.util.List;

import org.hibernate.SQLQuery;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
import org.hibernate.cfg.Configuration;

public class Client {

    private static SessionFactory factory;

    //Connects with the db. Creates sessionFactory object. With this object
    // you can create a session, i.e., a connection is established between client and server.
    public static void getSessionFactory() {

        try {
            Configuration conf = new Configuration().configure();

            StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder().applySettings(conf.getProperties());

            factory = conf.buildSessionFactory(builder.build());
        }catch (Exception ex) {
            System.out.println(ex);
        }
    }
}
```

```

32@    public static void main (String args[]) {
33        getSessionFactory();
34
35        Client clientObj = new Client();
36
37        //Inserts record in the db.
38        //clientObj.insertRecordInDatabase (75, "Soumyajit", "Mondal", 40000);
39
40        //Display all the records using HQL klientObj.displayRecords();
41
42        //Display the records using native Query.
43        //clientObj.displayRecordsUsing NativeQuery();
44
45        //Update the data in the db.
46        //clientObj.updateRecord(75, "Soumyajit");
47
48        //Delete record from the db.
49        //clientObj.deleteRecord (13);
50
51    }
52
53    //Inserts a record in the db.
54

//Display all the records using HQL (Hibernate Query Language)
public void displayRecords() {

    Session session = factory.openSession();

    //HQL - Hibernate Query Language..
    //Select * from hibernate_create_insert_update_delete
    List<Employee> empLst = session.createQuery("FROM Employee").list();

    for (Employee empObj : empLst) {
        System.out.println(empObj);
    }

    session.close();
}

//Displays the records using native SQL Query.
public void displayRecordsUsingNativeQuery() {
    Session session = factory.openSession();

    String str = "SELECT * from hibernate_create_insert_update_delete WHERE salary >= 10000";
    SQLQuery query = session.createSQLQuery(str);
    query.addEntity(Employee.class);

    List<Employee> empLst = query.list();

    for (Employee empObj : empLst)
        System.out.println(empObj);

    session.close();
}

```

```

//Update the record in the db.
public void updateRecord (int id, String firstName) {
    Session session = factory.openSession();
    Transaction tx = session.beginTransaction();

    //SELECT * FROM hibernate_create_insert_update_delete WHERE id = given id
    Employee empObj = (Employee) session.get(Employee.class, id);

    empObj.setFirstName(firstName);

    session.update(empObj);

    tx.commit();
    System.out.println("Given record is updated in the db successfully...");
    session.close();
}

//Deletes the record from the db.
public void deleteRecord (int id) {
    Session session = factory.openSession();
    Transaction tx = session.beginTransaction();

    Employee empObj = (Employee) session.get(Employee.class, id);
    session.delete(empObj);

    tx.commit();
    System.out.println("Record with id "+ id +" is deleted successfully");
    session.close();
}

```

Spring JPA

application.properties

```

1 spring.application.name=SpringBootSpringJPA
2 server.port=8091
3 spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
4 spring.datasource.url=jdbc:mysql://127.0.0.1:3306/test
5 spring.datasource.username=root
6 spring.datasource.password=admin
7 spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL8Dialect
8 spring.jpa.hibernate.ddl-auto=create
9 spring.jpa.show-sql=true
10

```

Employee.java

```
1 package com.soumya.demo.data;
2
3@import jakarta.persistence.Entity;[]
4
5 @Entity
6 @Table(name="Employee")
7
8 public class Employee {
9
10     @Id
11     int id;
12
13     String name;
14     String dept;
15     String designation;
16
17     Employee(){}
18
19
20     public Employee(int id, String name, String dept, String designation) {
21         super();
22         this.id = id;
23         this.name = name;
24         this.dept = dept;
25         this.designation = designation;
26     }
27
28
29     public int getId() {
30         return id;
31     }
32
33     public void setId(int id) {
34         this.id = id;
35     }
36
37     public String getName() {
38         return name;
39     }
40
41     public void setName(String name) {
42         this.name = name;
43     }
44
45     public String getDept() {
46         return dept;
47     }
48
49     public void setDept(String dept) {
50         this.dept = dept;
51     }
52
53     public String getDesignation() {
54         return designation;
55     }
56
57     public void setDesignation(String designation) {
58         this.designation = designation;
59     }
60
61
62 }
```

Activate Windows

Employee Repository

EmployeeRepository.java

```
1 package com.soumya.demo.repository;
2
3 import org.springframework.data.jpa.repository.JpaRepository;
4
5 public interface EmployeeRepository extends JpaRepository<Employee, Integer> {
6
7
8
9
10}
11}
12
```

Employee Controller

EmployeeController.java

```
1 package com.soumya.demo.controller;
2
3 import java.util.List;
4
5 @RestController
6 public class EmployeeController {
7
8     @Autowired
9     EmployeeRepository empRepository;
10
11     @GetMapping("getAllEmployee")
12     public List<Employee> getAllEmployees(){
13         List<Employee> empLst = empRepository.findAll();
14         return empLst;
15     }
16
17     //Insert the data
18     @PostMapping("/insertData")
19     public String insertData(@RequestBody Employee empObj) {
20         System.out.println("Given Object is : "+empObj);
21         empRepository.save(empObj);
22
23         String strReturn = "Record Inserted in Database Successfully";
24         return strReturn;
25     }
26
27
28
29
30
31
32
33
34
35
36
37
38 }
```

```

39  @PutMapping ("/updateData")
40  public String updateData(@RequestBody Employee empObj) {
41      System.out.println("Given Object to Update is : "+empObj);
42      empRepository.save(empObj);
43
44      String strReturn = "Record Updated in Database Successfully";
45      return strReturn;
46  }
47
48  @DeleteMapping("/deleteData/{id}")
49  public String deleteData(@PathVariable Employee empObj) {
50      System.out.println("Given Object to Delete is : "+empObj);
51      empRepository.delete(empObj);
52
53      String strReturn = "Record Deleted in Database Successfully";
54      return strReturn;
55  }
56
57 }

```

Activate Windows
Go to Settings to activate Windows

CRUD Operation

GET Data

```

localhost:8091/getAllEmployee
Pretty-print □
[{"id":2323233,"name":"Rahul Das","dept":"IT","designation":"Full Stack Developer"}]

```

POST _ Insert Data

POST <http://localhost:8091/insertData> Send

Params Auth Headers (8) Body ● Pre-req. Tests Settings Cookies

raw JSON Beautify

```

1 {
2     "id":2323233,
3     "name":"Soumyajit Mondal",
4     "dept":"IT",
5     "designation":"Full Stack Developer"

```

Body 200 OK 32 ms 204 B Save Response

Pretty Raw Preview Visualize Text

1 Record Inserted in Database Successfully

Activate Windows

PUT Data

The screenshot shows the Postman interface with a PUT request to `http://localhost:8091/updateData`. The request body is a JSON object:

```
1 "id":2323233,
2 "name":"Soumyajit",
3 "dept":"IT",
4 "designation":"Full Stack Developer"
```

The response status is 200 OK, 17 ms, 203 B. The response body is: "Record Updated in Database Successfully".

Delete Data

The screenshot shows the Postman interface with a DELETE request to `http://localhost:8091/deleteData/1234`. The request body is a JSON object:

```
1 "id":1234
```

The response status is 200 OK, 17 ms, 203 B. The response body is: "Record Deleted in Database Successfully".

Spring MicroServices

Eureka Server Enable

SpringEurekaServerApplication.java

```
1 package com.soumya.demo;
2
3 import org.springframework.boot.SpringApplication;
4
5 @SpringBootApplication
6 @EnableEurekaServer
7 public class SpringEurekaServerApplication {
8
9     public static void main(String[] args) {
10         SpringApplication.run(SpringEurekaServerApplication.class, args);
11     }
12 }
```

application.properties

```
1 spring.application.name=SpringEurekaServer
2 server.port=8761
3 eureka.client.register-with-eureka=false
4 eureka.client.fetch-registry=false
5
```

Server is running or not

The screenshot shows the Spring Eureka dashboard at localhost:8761. The top navigation bar includes links for HOME and LAST 1000 SINCE STARTUP, along with user and settings icons.

System Status

Environment	test	Current time	2024-05-08T23:30:17 +0530
Data center	default	Uptime	00:00
		Lease expiration enabled	false
		Renews threshold	1
		Renews (last min)	0

DS Replicas

Instances currently registered with Eureka

Activate Windows
Go to Settings to activate Windows.

Eureka Client Employee

application.properties

```
1 spring.application.name=SpringEurekaClientEmployeeData1
2 server.port=8082
3 eureka.client.service-url.defaultZone=http://localhost:8761/eureka
4 eureka.client.healthcheck.enabled=true
5 eureka.client.fetch-registry=true
6
7
```

Enable Discovery Client

```
1 package com.soumya.demo;
2
3@import org.springframework.boot.SpringApplication;
4
5 @SpringBootApplication
6 @EnableDiscoveryClient
7 public class SpringEurekaClientEmployeeData1Application {
8
9     public static void main(String[] args) {
10         SpringApplication.run(SpringEurekaClientEmployeeData1Application.class, args);
11     }
12 }
13
14
15 }
```

MyController

```
1 package com.soumya.demo.controller;
2
3@import org.springframework.web.bind.annotation.GetMapping;
4
5 @RestController
6 public class MyController {
7
8     @GetMapping("/getDataWithID/{id}")
9     public String getDataWithID(@PathVariable int id) {
10         System.out.println("Received ID from Client is : "+ id);
11         if(id == 101)
12             return "Soumya";
13         else if(id == 102)
14             return "Charan Sir";
15         else if (id == 103)
16             return "Sakshi";
17
18         else if (id == 104)
19             return "Ankita";
20         else if (id == 105)
21             return "Trupti";
22
23         return "Invalid Employee";
24     }
25
26 }
27
28 }
```

Run the program

The screenshot shows the Eureka dashboard at localhost:8761. At the top, there are two status indicators: "Renews threshold" set to 5 and "Renews (last min)" set to 2. A prominent red warning message states: "EMERGENCY! EUREKA MAY BE INCORRECTLY CLAIMING INSTANCES ARE UP WHEN THEY'RE NOT. RENEWALS ARE LESSER THAN THRESHOLD AND HENCE THE INSTANCES ARE NOT BEING EXPIRED JUST TO BE SAFE." Below this, the "DS Replicas" section lists registered instances:

Application	AMIs	Availability Zones	Status
SPRINGEUREKACLIENTEMPLOYEE DATA1	n/a (1) (1)		UP (1) - localhost:SpringEurekaClientEmployeeData1:8082
SPRINGEUREKACLIENTPAYROLL	n/a (1) (1)		UP (1) - localhost:SpringEurekaClientPayroll:8081

Below the instance list is a "General Info" section with a link to "Activate Windows".

Client PayRoll

Application.properties

```
1 spring.application.name=SpringEurekaClientPayroll
2 server.port=8081
3 eureka.client.service-url.defaultZone=http://localhost:8761/eureka
4 eureka.client.healthcheck.enabled=true
5 eureka.client.fetch-registry=true
6
```

ClientPayrollApplication.java

```
1 package com.soumya.demo;
2
3 import org.springframework.boot.SpringApplication;
4
5 @SpringBootApplication
6 @EnableDiscoveryClient
7 public class SpringEurekaClientPayrollApplication {
8
9     public static void main(String[] args) {
10         SpringApplication.run(SpringEurekaClientPayrollApplication.class, args);
11     }
12
13     @Bean
14     @LoadBalanced
15     public RestTemplate restTemplate(RestTemplateBuilder builder) {
16         return builder.build();
17     }
18
19 }
```

MyController

```
1 package com.soumya.demo.controller;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4
5 @RestController
6 public class EmployeeController {
7
8     @Autowired
9     RestTemplate restTemplate;
10
11     @GetMapping ("/getEmployeeData/{id}")
12     public String getEmployeeData(@PathVariable int id) {
13         System.out.println("Obtained ID from the User is : " + id);
14
15         String strURL = "http://EMPLOYEE/getDataWithID/" + id;
16         String str = restTemplate.getForObject(strURL, String.class);
17         return str;
18     }
19
20 }
```

Client-Registered

localhost:8761

Lease expiration enabled	false
Renews threshold	5
Renews (last min)	2

EMERGENCY! EUREKA MAY BE INCORRECTLY CLAIMING INSTANCES ARE UP WHEN THEY'RE NOT. RENEWALS ARE LESSER THAN THRESHOLD AND HENCE THE INSTANCES ARE NOT BEING EXPIRED JUST TO BE SAFE.

DS Replicas

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
SPRINGEUREKACLIENTEMPLOYEEDATA1	n/a (1) (1)		UP (1) - localhost:SpringEurekaClientEmployeeData1:8082
SPRINGEUREKACLIENTPAYROLL	n/a (1) (1)		UP (1) - localhost:SpringEurekaClientPayroll:8081

Spring Reactor

testReactor

testReactor.java

```
1 package testReactor;
2
3 import reactor.core.publisher.Mono;
4
5 public class testReactor {
6
7     public Mono<String> nameMono(){
8         return Mono.just("Soumyajit Mondal").log();
9     }
10
11    public static void main(String[] args) {
12        testReactor testObj = new testReactor();
13        testObj.nameMono().subscribe(name -> System.out.println("My name is " + name));
14    }
15
16 }
```

Output

```
<terminated> testReactor [Java Application] C:\software\sts-4.22.0.RELEASE\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_
[ INFO] (main) | onSubscribe([Synchronous Fuseable] Operators.ScalarSubscription)
[ INFO] (main) | request(unbounded)
[ INFO] (main) | onNext(Soumyajit Mondal)
My name is Soumyajit Mondal
[ INFO] (main) | onComplete()
```

SpringReactorMono

SpringReactorMonoApplication.java

```
1 package com.soumya.demo;
2
3 import org.springframework.boot.SpringApplication;
4
5 @SpringBootApplication
6 public class SpringReactorMonoApplication {
7
8     public static void main(String[] args) {
9         SpringApplication.run(SpringReactorMonoApplication.class, args);
10    }
11
12 }
13
14 }
```

MyController

```
1 package com.soumya.demo.controller;
2
3@import org.springframework.web.bind.annotation.GetMapping;
4 import org.springframework.web.bind.annotation.RestController;
5
6 import reactor.core.publisher.Mono;
7
8 @RestController
9 public class MyController {
10
11@    @GetMapping("getName")
12    public Mono<String> getName(){
13        return Mono.just("My name is Soumyajit Mondal").log();
14    }
15}
16
17}
18
```

Output

