JSF - DISPLAY DATATABLE

h:dataTable tag is used to display data in tabular fashion.

JSF Tag

```
<h:dataTable value="#{userData.employees}" var="employee"
   styleClass="employeeTable"
   headerClass="employeeTableHeader"
   rowClasses="employeeTableOddRow,employeeTableEvenRow">
   <h:column>
      <f:facet name="header">Name</f:facet>
      #{employee.name}
   </h:column>
   <h:column>
      <f:facet name="header">Department</f:facet>
      #{employee.department}
   </h:column>
   <h:column>
      <f:facet name="header">Age</f:facet>
      #{employee.age}
   </h:column>
   <h:column>
      <f:facet name="header">Salary</f:facet>
      #{employee.salary}
   </h:column>
</h:dataTable>
```

Rendered Output

```
<table >
<thead><tr>
   <th >Name</th>
   <th >Department</th>
   <th >Age</th>
   <th >Salary</th>
</tr></thead>
<tbody>
<tr >
   <td>John</td>
   <td>Marketing</td>
   <td>30</td>
   <td>2000.0</td>
</tr>
<tr >
   <td>Robert</td>
   <td>Marketing</td>
   <td>35</td>
   <td>3000.0</td>
</tr>
</table>
```

Tag Attributes

S.N.	Attribute & Description
1	**id** Identifier for a component
2	**rendered**

A boolean; false suppresses rendering

3 **dir**

Direction for text. Valid values are **ltr** *lefttoright* and **rtl** *righttoleft*.

4 **styleClass**

Cascading stylesheet *CSS* class name

5 **value**

A component's value, typically a value binding

6 **bgcolor**

Background color for the table

7 **border**

Width of the table's border

8 **cellpadding**

Padding around table cells

9 **cellspacing**

Spacing between table cells

10 **columnClasses**

Comma-separated list of CSS classes for columns

11 **first**

Index of the first row shown in the table

12 **footerClass**

CSS class for the table footer

13 **frame**

Specification for sides of the frame surrounding the table should be drawn; valid values: none, above, below, hsides, vsides, lhs, rhs, box, border

14 **headerClass**

CSS class for the table header

15 **rowClasses**

Comma-separated list of CSS classes for rows

16 **rules**

Specification for lines drawn between cells; valid values: groups, rows, columns, all

17 **summary**

Summary of the table's purpose and structure used for non-visual feedback such as speech

18 **var**

The name of the variable created by the data table that represents the current item in the value

19 **title**

A title, used for accessibility, that describes an element. Visual browsers typically create tooltips for the title’s value

20 **width**

Width of an element

21 **onblur**

Element loses focus

22 **onchange**

Element’s value changes

23 **onclick**

Mouse button is clicked over the element

24 **ondblclick**

Mouse button is double-clicked over the element

25 **onfocus**

Element receives focus

26 **onkeydown**

Key is pressed

27 **onkeypress**

Key is pressed and subsequently released

28 **onkeyup**

Key is released

29 **onmousedown**

Mouse button is pressed over the element

30 **onmousemove**

Mouse moves over the element

31 **onmouseout**

Mouse leaves the element's area

32 **onmouseover**

Mouse moves onto an element

33 **onmouseup**

Mouse button is released

Example Application

Let us create a test JSF application to test the above tag.

Step	Description
1	Create a project with a name *helloworld* under a package *com.tutorialspoint.test* as explained in the *JSF - h:outputStylesheet* sub-chapter of *JSF - Basic Tags* chapter.
2	Modify *styles.css* as explained below.
3	Create *Employee.java* under package *com.tutorialspoint.test* as explained below.
4	Create *UserData.java* as a managed bean under package *com.tutorialspoint.test* as explained below.
5	Modify *home.xhtml* as explained below. Keep rest of the files unchanged.
6	Compile and run the application to make sure business logic is working as per the requirements.
7	Finally, build the application in the form of war file and deploy it in Apache Tomcat Webserver.
8	Launch your web application using appropriate URL as explained below in the last step.

styles.css

```
.employeeTable{
   border-collapse:collapse;
   border:1px solid #000000;
}

.employeeTableHeader{
   text-align:center;
   background:none repeat scroll 0 0 #B5B5B5;
   border-bottom:1px solid #000000;
   padding:2px;
}

.employeeTableOddRow{
   text-align:center;
   background:none repeat scroll 0 0 #FFFFFF;
```

```
}

.employeeTableEvenRow{
   text-align:center;
   background:none repeat scroll 0 0 #D3D3D3;
}
```

Employee.java

```
package com.tutorialspoint.test;

public class Employee {
   private String name;
   private String department;
   private int age;
   private double salary;
   private boolean canEdit;

   public Employee (String name,String department,int age,double salary){
      this.name = name;
      this.department = department;
      this.age = age;
      this.salary = salary;
      canEdit = false;
   }

   public String getName() {
      return name;
   }

   public void setName(String name) {
      this.name = name;
   }

   public String getDepartment() {
      return department;
   }

   public void setDepartment(String department) {
      this.department = department;
   }

   public int getAge() {
      return age;
   }

   public void setAge(int age) {
      this.age = age;
   }

   public double getSalary() {
      return salary;
   }

   public void setSalary(double salary) {
      this.salary = salary;
   }

   public boolean isCanEdit() {
      return canEdit;
   }

   public void setCanEdit(boolean canEdit) {
      this.canEdit = canEdit;
   }
}
```

UserData.java

```java
package com.tutorialspoint.test;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.Arrays;

import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;

@ManagedBean(name = "userData", eager = true)
@SessionScoped
public class UserData implements Serializable {

private static final long serialVersionUID = 1L;

   private String name;
   private String dept;
   private int age;
   private double salary;

   private static final ArrayList<Employee> employees
      = new ArrayList<Employee>(Arrays.asList(
      new Employee("John", "Marketing", 30,2000.00),
      new Employee("Robert", "Marketing", 35,3000.00),
      new Employee("Mark", "Sales", 25,2500.00),
      new Employee("Chris", "Marketing", 33,2500.00),
      new Employee("Peter", "Customer Care", 20,1500.00)
   ));


   public ArrayList<Employee> getEmployees() {
      return employees;
   }

   public String addEmployee() {
      Employee employee = new Employee(name,dept,age,salary);
      employees.add(employee);
      return null;
   }

   public String deleteEmployee(Employee employee) {
      employees.remove(employee);
      return null;
   }

   public String editEmployee(Employee employee){
      employee.setCanEdit(true);
      return null;
   }

   public String saveEmployees(){
      //set "canEdit" of all employees to false
      for (Employee employee : employees){
         employee.setCanEdit(false);
      }
      return null;
   }

   public String getName() {
      return name;
   }

   public void setName(String name) {
      this.name = name;
   }

   public String getDepartment() {
      return department;
```

```
    }

    public void setDepartment(String department) {
        this.department = department;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public double getSalary() {
        return salary;
    }

    public void setSalary(double salary) {
        this.salary = salary;
    }
}
```

home.xhtml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
xmlns:h="http://java.sun.com/jsf/html"
xmlns:f="http://java.sun.com/jsf/core">
    <h:head>
        <title>JSF tutorial</title>
        <h:outputStylesheet library="css" name="styles.css"  />
    </h:head>
    <h:body>
    <h2>DataTable Example</h2>
    <h:form>
        <h:dataTable value="#{userData.employees}" var="employee"
            styleClass="employeeTable"
            headerClass="employeeTableHeader"
            rowClasses="employeeTableOddRow,employeeTableEvenRow">
            <h:column>
                <f:facet name="header">Name</f:facet>
                #{employee.name}
            </h:column>
            <h:column>
                <f:facet name="header">Department</f:facet>
                #{employee.department}
            </h:column>
            <h:column>
                <f:facet name="header">Age</f:facet>
                #{employee.age}
            </h:column>
            <h:column>
                <f:facet name="header">Salary</f:facet>
                #{employee.salary}
            </h:column>
        </h:dataTable>
    </h:form>
    </h:body>
</html>
```

Once you are ready with all the changes done, let us compile and run the application as we did in JSF - First Application chapter. If everything is fine with your application, this will produce following result: