ICON OF JAVA

ADV. JAVA

JSP

11. JSP Standard Tag Library (JSTL)

Mr. Nagoor Babu M.Tech

[ICON of JAVA]

(Sun certified & Realtime Expert)

Ex. HCL Employee

Trained Lakhs of Students
for last 14 years across INDIA

India's No.1 Software Training Institute

DURGASOFT

www.durgasoft.com Ph: 9246212143 ,8096969696

JSP Standard Tag Library (JSTL)

In Jsp technology, by using scripting elements we are able to provide Java code inside the Jsp pages.

To preserve Jsp principles we have to eliminate scripting elements, for this we have to use Jsp Actions.

In case of Jsp Actions, we will use standard actions as an alternative to scripting elements, but which are limited in number and having bounded functionality so that standard actions are not specified the required application format.

Still it is required to provide Java code inside the Jsp pages.

In the above context, to eliminate Java code completely from Jsp pages we have to use custom actions.

In case of custom actions, to implement simple Java syntaxes like if condition, for loop and so on we have to provide a lot of Java code internally.

To overcome the above problem Jsp technology has provided a separate tag library with simple Java syntaxes implementation and frequently used operations.

JSTL is an abstraction provided by Sun Microsystems, but where implementations are provided by all the server vendors.

With the above convention Apache Tomcat has provided JSTL implementation in the form of the jar files as standard.jar, jstl.jar.

Apache Tomcat has provided the above 2 jar files in the following location.

C:\Tomcat7.0\webapps\examples\WEB_INF\lib

If we want to get JSTL support in our Jsp pages then we have to keep the above 2 jar files in our web application lib folder.

JSTL has provided the complete tag library in the form of the following 5 types of tags.

1. Core Tags
2. XML Tags
3. Internationalization or I18N Tags (Formatted tags)
4. SQL Tags
5. Functions tags

To get a particular tag library support into the present Jsp page we have to use the following standard URL's to the attribute in the respective taglib directive.

http://java.sun.com/jstl/core

http://java.sun.com/jstl/xml

http://java.sun.com/jstl/fmt

http://java.sun.com/jstl/sql

http://java.sun.com/jsp/jstl/functions

1. Core Tags:

In JSTL, core tag library was divided into the following 4 types.

1. General Purpose Tags
    1. <c:set----->
    2. <c:remove----->
    3. <c:catch----->
    4. <c:out----->
2. Conditional Tags
    1. <c:if----->
    2. <c:choose----->
    3. <c:when----->
    4. <c:otherwise----->
3. Iterate Tags

1. <c:forEach----->
2. <c:forTokens----->

4. Url-Based Tags
    1. <c:import----->
    2. <c:url----->
    3. <c:redirect----->

1. General Purpose Tags:

1. <c:set----->:

This tag can be used to declare a single name value pair onto the specified scope.

Syntax:<c:set var="--" value="--" scope="--"/>

Where var attribute will take a variable i.e. key in key-value pair.

Where value attribute will take a particular value i.e. a value in key-value pair.

Where scope attribute will take a particular scope to include the specified key-value pair.

2. <c:out----->:

This tag can be used to display a particular value on the client browser.

Syntax:<c:out value="--"/>

Where value attribute will take a static data or an expression.

To present an expression with value attribute we have to use the following format.

Syntax:${expression}

Ex:<c:out value"${a}"/>

If the container encounters above tag then container will search for 'a' attribute in the page scope, request scope, session scope and application scope.

----------Application13----------

jstlapp1:

core.jsp:

```
<%@taglib uri="http://java.sun.com/jstl/core" prefix="c"%>
<%@page isELIgnored="true"%>
<html>
	<body>
		<center><b><font size="7">
		<c:set var="a" value="AAA" scope="request"/>
		<br>
		<c:out value="core tag library"/>
		<br><br>
		<c:out value="${a}"/>
		</font></b></center>
	<body>
<html>
```

3. <c:remove----->:

This tag can be used to remove an attribute from the specified scope.

Syntax:<c:remove var="--" scope="--"/>

Where scope attribute is optional, if we have not specified scope attribute then container will search for the respective attribute in the page scope, request scope, session scope and

application scope.

----------Application14----------

core1.jsp:

```
<%@taglib uri="http://java.sun.com/jstl/core" prefix="c"%>
<%@page isELIgnored="true"%>
<html>
	<body>
		<center><b><font size="7">
		<c:set var="a" value="AAA" scope="request"/>
		<br>
		a-----><c:out value="${a}"/>
		<br><br>
		<c:remove var="a" scope="request"/>
		a-----><c:out value="${a}"/>
		</font></b></center>
	<body>
<html>
```

4. <c:catch----->:

This tag can be used to catch an Exception raised in its body.

Syntax:<c:catch var="--">

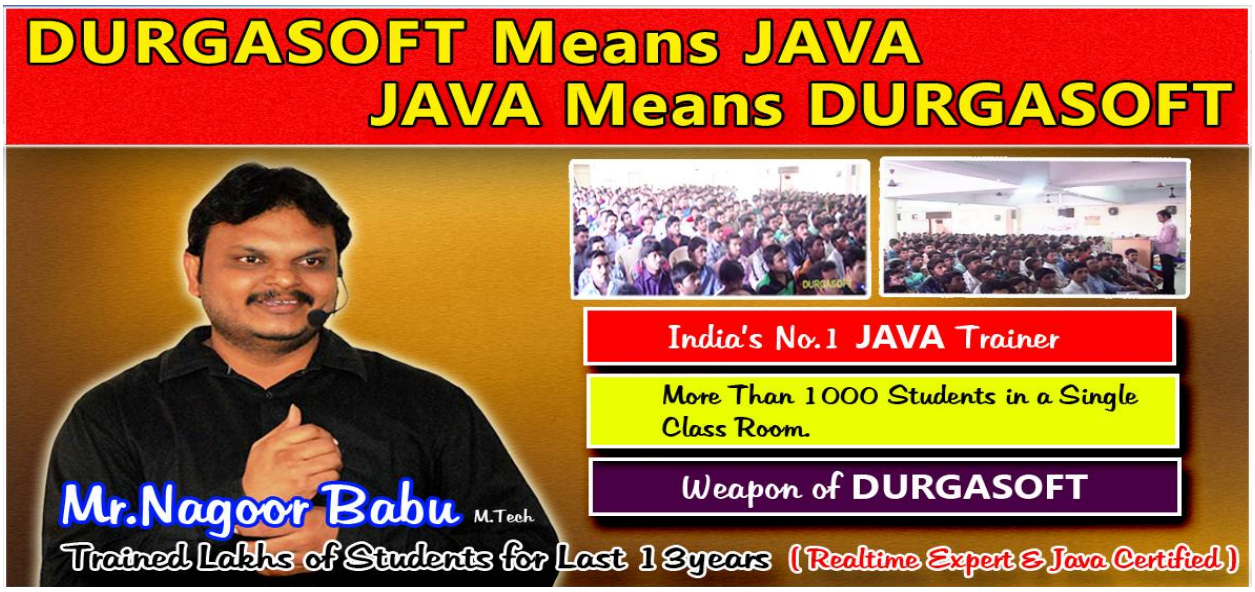
---------- </c:catch>

Where var attribute will take a variable to hold the generated Exception object reference.

----------Application15----------

core2.jsp:

```
<%@taglib uri="http://java.sun.com/jstl/core" prefix="c"%>
<%@page isELIgnored="true"%>
<html>
	<body>
		<center><b><font size="7">
		<c:catch var="e">
			<jsp:scriptlet>
				java.util.Date d=null;
				out.println(d.toString());
			</jsp:scriptlet>
		</c:catch>
		<c:out value="${e}"/>
		</font></b></center>
	<body>
<html>
```



2. Conditional Tags:

1. <c:if----->:

This tag can be used to implement if conditional statement.

Syntax:<c:if test="--"/>

Where test attribute is a boolean attribute, it may take either true or false values.

----------Application16----------

core3.jsp:

```
<%@taglib uri="http://java.sun.com/jstl/core" prefix="c"%>
<%@page isELIgnored="true"%>
<html>
	<body>
		<center><b><font size="7">
			<c:set var="a" value="10"/>
			<c:set var="b" value="20"/>
			<c:if test="${a<b}">
				condition is true
			</c:if>
			<br>
			out of if
		</font></b></center>
	<body>
<html>
```

2. <c:choose----->, <c:when-----> and <c:otherwise----->:

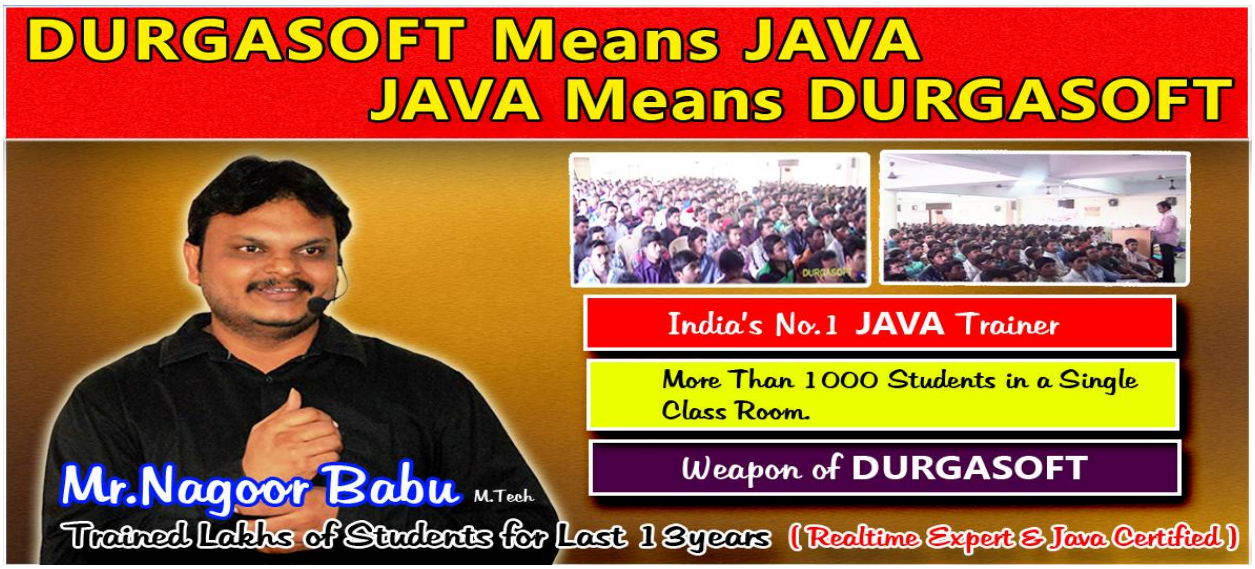
These tags can be used to implement switch programming construct.

Syntax:
```
<c:choose>
<c:when test="--">
            ----------
</c:when>
            ----------
<c:otherwise>
            ----------
</c:otherwise>
</c:choose>
```

----------Application17----------

core4.jsp:

```
<%@taglib uri="http://java.sun.com/jstl/core" prefix="c"%>
<%@page isELIgnored="true"%>
<html>
      <body>
            <center><b><font size="7">
                  <c:set var="a" value="10"/>
                  <c:choose>
                        <c:when test="${a==10}">
                              TEN
                        </c:when>
                        <c:when test="${a==15}">
                              FIFTEEN
                        </c:when>
                        <c:when test="${a==20}">
                              TWENTY
                        </c:when>
                        <c:otherwise>
                              Number is not in 10,15 and 20
                        </c:otherwise>
                  </c:choose>
            </font></b></center>
      <body>
<html>
```



3. Iterator Tags:

1. <c:forEach----->:

This tag can be used to implement for loop to provide iterations on its body and it can be used to perform iterations over an array of elements or Collection of elements.

Syntax 1:<c:forEach var="--" begin="--" end="--" step="--">

</c:forEach>

Where var attribute will take a variable to hold up the loop index value at each and every iteration.

Where begin and end attribute will take start index value and end index value.

Syntax 2:<c:forEach var="--" items="--">

</c:forEach>

Where var attribute will take a variable to hold up an element from the respective Collection at each and every iteration.

Where items attribute will take the reference of an array or Collection from either of the scopes page, request, session and application.

----------Application18----------

core5.jsp:

```
<%@taglib uri="http://java.sun.com/jstl/core" prefix="c"%>
<%@page isELIgnored="true"%>
<html>
	<body>
		<center><b><font size="7">
			<c:forEach var="a" begin="0" end="10" step="2">
				<c:out value="${a}"/><br>
			</c:forEach>
			<%
				String[] s={"A","B","C"};
				request.setAttribute("s",s);
			%>
			<br>
			<c:forEach var="x" items="${s}">
				<c:out value="${x}"/><br>
			</c:forEach>
		</font></b></center>
	<body>
<html>
```

2. <c:forTokens----->:

This tag can be used to perform String Tokenization on a particular String.

Syntax :<c:forTokens var="--" items="--" delims="--">

```
	------------

</c:forTokens>
```

Where var attribute will take a variable to hold up token at each and every iteration.

Where items attribute will take a String to tokenize.

Where delims attribute will take a delimiter to perform tokenization.

----------Application19----------

core6.jsp:

```
<%@taglib uri="http://java.sun.com/jstl/core" prefix="c"%>
<%@page isELIgnored="true"%>
<html>
	<body>
		<center><b><font size="7">
			<c:forTokens var="token" items="Durga Software Solutions" delims="
">
				<c:out value="${token}"/><br>
			</c:forTokens>
```

```
            </font></b></center>
        <body>
<html>
```

4. Url-Based Tags:

1. <c:import----->:

This tag can be used to import the content of the specified target resource into the present Jsp page.

----------Application20----------

second.jsp:

```
<center><h1>This is second.jsp</h1></center>
```

core7.jsp:

```
<%@taglib uri="http://java.sun.com/jstl/core" prefix="c"%>

<%@page isELIgnored="true"%>
<html>
        <body>
                <center><b><font size="7">
                        Start
                        <br>
                        <c:import url="second.jsp"/>
                        <br>
                        End
                </font></b></center>
        <body>
<html>
```

2. <c:url----->:

This tag can be used to represent the specified url.

Syntax :<c:url value="--"/>

----------Application21----------

core8.jsp:

```
<%@taglib uri="http://java.sun.com/jstl/core" prefix="c"%>
<%@page isELIgnored="true"%>
<html>
	<body>
		<center><b><font size="7">
			<a href='<c:url value="http://localhost:2020/loginapp"/>'>Login
Application</a>
		</font></b></center>
	<body>
<html>
```

2. <c:redirect----->:

This tag can be used to implement Send Redirect Mechanism from a particular Jsp page.

Syntax:<c:redirect url="--"/>

----------Application22----------

core9.jsp:

```
<%@taglib uri="http://java.sun.com/jstl/core" prefix="c"%>
<%@page isELIgnored="true"%>

<html>
```

```
        <body>
                <center><b><font size="7">
                        <c:redirect url="http://localhost:2020/registrationapp"/>
                </font></b></center>
        <body>
<html>
```

4. SQL Tags:

The main purpose of SQL tag library is to interact with the database in order to perform the basic database operations.

JSTL has provided the following set of tags as part of SQL tag library.

1. <sql:setDataSource----->
2. <sql:update----->
3. <sql:query----->
4. <sql:transaction----->
5. <sql:param----->
6. <sql:dateParam----->

1. <sql:setDataSource----->:

This tag can be used to prepare the Jdbc environment like Driver loading, establish the connection.

Syntax:<sql:setDataSource driver="--" url="--" user="--" password="--"/>

Where driver attribute will take the respective Driver class name.

Where url attribute will take the respective Driver url.

Where user and password attributes will take database user name and password.

2. <sql:update----->:

This tag can be used to execute all the updation group SQL queries like create, insert, update, delete, drop and alter.

Syntax 1:<sql:update var="--" sql="--"/>

Syntax 2:<sql:update var="--"> ------ query ------ </sql:update>

In the above <sql:update> tag we are able to provide the SQL queries in 2 ways.

1. Statement style SQL queries, which should not have positional parameters.
2. PreparedStatement style SQL queries, which should have positional parameters.

If we use PreparedStatement style SQL queries then we have to provide values to the positional parameters, for this we have to use the following SQL tags.

1. <sql:param----->:

This tag can be used to set a normal value to the positional parameter.

Syntax 1:<sql:param value="--"/>

Syntax 2:<sql:param> ------ value ------ </sql:param>

2. <sql:dateParam----->:

This tag can be used to set a value to parameter representing data.

Syntax 1:<sql:dateParam value="--"/>

Syntax 2:<sql:dateParam>value</sql:dateParam>

----------Application23----------

jstlapp2:

sql.jsp:

```
<%@taglib uri="http://java.sun.com/jstl/core" prefix="c"%>
<%@taglib uri="http://java.sun.com/jstl/sql" prefix="sql"%>
<%@page isELIgnored="true"%>
<html>
	<body>
		<center><b><font size="7">
			<sql:setDataSource driver="oracle.jdbc.driver.OracleDriver"
url="jdbc:oracle:thin:@localhost:1521:xe" user="system" password="durga"/>
			<sql:update var="result" sql="create table emp1(eid number, ename
varchar2(10), esal number)"/>
			Row Count ..... <c:out value="${result}"/>
		</font></b></center>
	</body>
</html>
```

----------Application24----------

sql1.jsp:

```
<%@taglib uri="http://java.sun.com/jstl/core" prefix="c"%>
<%@taglib uri="http://java.sun.com/jstl/sql" prefix="sql"%>
<%@page isELIgnored="true"%>
<html>
	<body>
		<center><b><font size="7">
			<sql:setDataSource driver="oracle.jdbc.driver.OracleDriver"
url="jdbc:oracle:thin:@localhost:1521:xe" user="system" password="durga"/>
			<sql:update var="result" sql="insert into emp1
values(101,'aaa',5000)"/>

			Row Count ..... <c:out value="${result}"/>
		</font></b></center>
	</body>
</html>
```

----------Application25----------

sql2.jsp:

```
<%@taglib uri="http://java.sun.com/jstl/core" prefix="c"%>
<%@taglib uri="http://java.sun.com/jstl/sql" prefix="sql"%>
<%@page isELIgnored="true"%>
<html>
	<body>
		<center><b><font size="7">
			<sql:setDataSource driver="oracle.jdbc.driver.OracleDriver"
url="jdbc:oracle:thin:@localhost:1521:xe" user="system" password="durga"/>
			<sql:update var="result" sql="insert into emp1 values(?,?,?)">
				<sql:param value="103"/>
				<sql:param>ccc</sql:param>
				<sql:param value="7000"/>
			</sql:update>
			Row Count ..... <c:out value="${result}"/>
		</font></b></center>
	</body>
</html>
```

----------Application26----------

sql3.jsp:

```
<%@taglib uri="http://java.sun.com/jstl/core" prefix="c"%>
<%@taglib uri="http://java.sun.com/jstl/sql" prefix="sql"%>
<%@page isELIgnored="true"%>
<html>
	<body>
		<center><b><font size="7">
			<sql:setDataSource driver="oracle.jdbc.driver.OracleDriver"
url="jdbc:oracle:thin:@localhost:1521:xe" user="system" password="durga"/>
			<sql:update var="result">
				update emp1 set esal=esal+? where esal>?
				<sql:param>500</sql:param>
				<sql:param>5000</sql:param>
			</sql:update>
			Row Count ..... <c:out value="${result}"/>


</font></b></center></body>
</html>
```

----------Application27----------

sql4.jsp:

```
<%@taglib uri="http://java.sun.com/jstl/core" prefix="c"%>
<%@taglib uri="http://java.sun.com/jstl/sql" prefix="sql"%>
<%@page isELIgnored="true"%>
<html>
	<body>
		<center><b><font size="7">
			<sql:setDataSource driver="oracle.jdbc.driver.OracleDriver"
url="jdbc:oracle:thin:@localhost:1521:xe" user="system" password="durga"/>
			<sql:update var="result">
				delete emp where esal>1000
			</sql:update>
			Row Count ..... <c:out value="${result}"/>
		</font></b></center>
	</body>
</html>
```

3. <sql:query----->:

This tag can be used to execute selection group SQL queries in order to fetch the data from database table.

Syntax 1:<sql:query var="--" sql="--"/>

Syntax 2:<sql:query var="--"> ----- query ----- </sql:query>

If we execute selection group SQL queries by using <sql:query> tag then SQL tag library will prepare result object to hold up fetched data.

In SQL tag library, result object is a combination of ResultSet object and ResultSetMetaData object.

In result object, all the column names will be represented in the form a single dimensional array referred by columnNames predefined variable and column data (table body) will be represented in the form of 2-dimensionnal array referred by rowsByIndex predefined variable.

----------Application28----------

sql5.jsp:

```
<%@taglib uri="http://java.sun.com/jstl/core" prefix="c"%>

<%@taglib uri="http://java.sun.com/jstl/sql" prefix="sql"%>
<%@page isELIgnored="true"%>
<html>
	<body>
		<center><b><font size="7">
			<sql:setDataSource driver="oracle.jdbc.driver.OracleDriver"
url="jdbc:oracle:thin:@localhost:1521:xe" user="system" password="durga"/>
			<sql:query var="result" sql="select * from emp"/>
			<table border="1" bgcolor="lightyellow">
			<tr>
				<c:forEach var="columnName"
items="${result.columnNames}">
					<td><center><b><font size="6" color="red">
					<c:out value="${columnName}"/>
					</font></b></center></td>
				</c:forEach>
			</tr>
			<c:forEach var="row" items="${result.rowsByIndex}">
				<tr>
					<c:forEach var="column" items="${row}">
						<td><b><font size="5">
							<c:out value="${column}"/>
						</font></b></td>
					</c:forEach>
				</tr>
			</c:forEach>
```

```
                    </table>
                </font></b></center>
        </body>
</html>
```

4. <sql:transaction----->:

This tag will represent a transaction, which includes collection of <sql:update> tags and <sql:query> tags.

3. I18N Tags(Formatted Tags):

1. <fmt:setLocale----->:

This tag can be used to represent a particular Locale.

Syntax:<fmt:setLocale value="--"/>

Where value attribute will take Locale parameters like en_US, it_IT and so on.

2. <fmt:formatNumber----->:

This tag can be used to represent a number w.r.t the specified Locale.

Syntax:<fmt:formatNumber var="--" value="--"/>

Where var attribute will take a variable to hold up the formatted number.

Where value attribute will take a number.

3. <fmt:formatDate----->:

This tag can be used to format present system date w.r.t. a particular Locale.

Syntax:<fmt:formatDate var="--" value="--"/>

Where var attribute will take a variable to hold up the formatted date.

Where value attribute will take the reference of Date object.

4. <fmt:setBundle----->:

This tag can be used to prepare ResourceBundle object on the basis of a particular properties file.

Syntax:<fmt:setBundle var="--" basename="--"/>

Where var attribute will take a variable to hold up ResourceBundle object reference.

Where basename attribute will take base name of the properties file.

5. <fmt:message----->:

This tag can be used to get the message from ResourceBundle object on the basis of provided key.

Syntax:<fmt:message var="--" key="--"/>

Where var attribute will take a variable to hold up message.

Where key attribute will take key of the message defined in the respective properties file.

----------Application29----------

jstlapp3:

abc_en_US.properties:

```
#abc_en_US.properties
#------------------------------
welcome=Welcome to US user
```

abc_it_IT.properties:

```
#abc_it_IT.properties
#------------------------------
welcome=Welcome toe Italiano usereo
```

fmt.jsp:

```
<%@taglib uri="http://java.sun.com/jstl/core" prefix="c"%>

<%@taglib uri="http://java.sun.com/jstl/fmt" prefix="fmt"%>
<%@page isELIgnored="true"%>
<html>
	<body>
		<center><b><font size="7">
			<fmt:setLocale value="it_IT"/>
			<fmt:formatNumber var="num" value="123456.789"/>
			<c:out value="${num}"/><br><br>
			<jsp:useBean id="date" class="java.util.Date">
				<fmt:formatDate var="fdate" value="${date}"/>
				<c:out value="${fdate}"/>
			</jsp:useBean>
			<br><br>
```

```
                    <fmt:setBundle basename="abc"/>
                    <fmt:message var="msg" key="welcome"/>
                    <c:out value="${msg}"/>
                </font></b></center>
        </body>
</html>
```

JSTL Functions:

The main purpose of functions tag library is to perform all the String operations which are defined in the String class.

Ex:<%taglib uri="http://java.sun.com/jstl/core" prefix="c"%>

<%taglib uri="http://java.sun.com/jstl//jspfunctions" prefix="fn"%>

<c:set var="a" value="Durga Software Solutions"/>

${fn.length(a)}

${fn.concat(a, "Hyderabad")}

${fn.toLowerCase(a)}

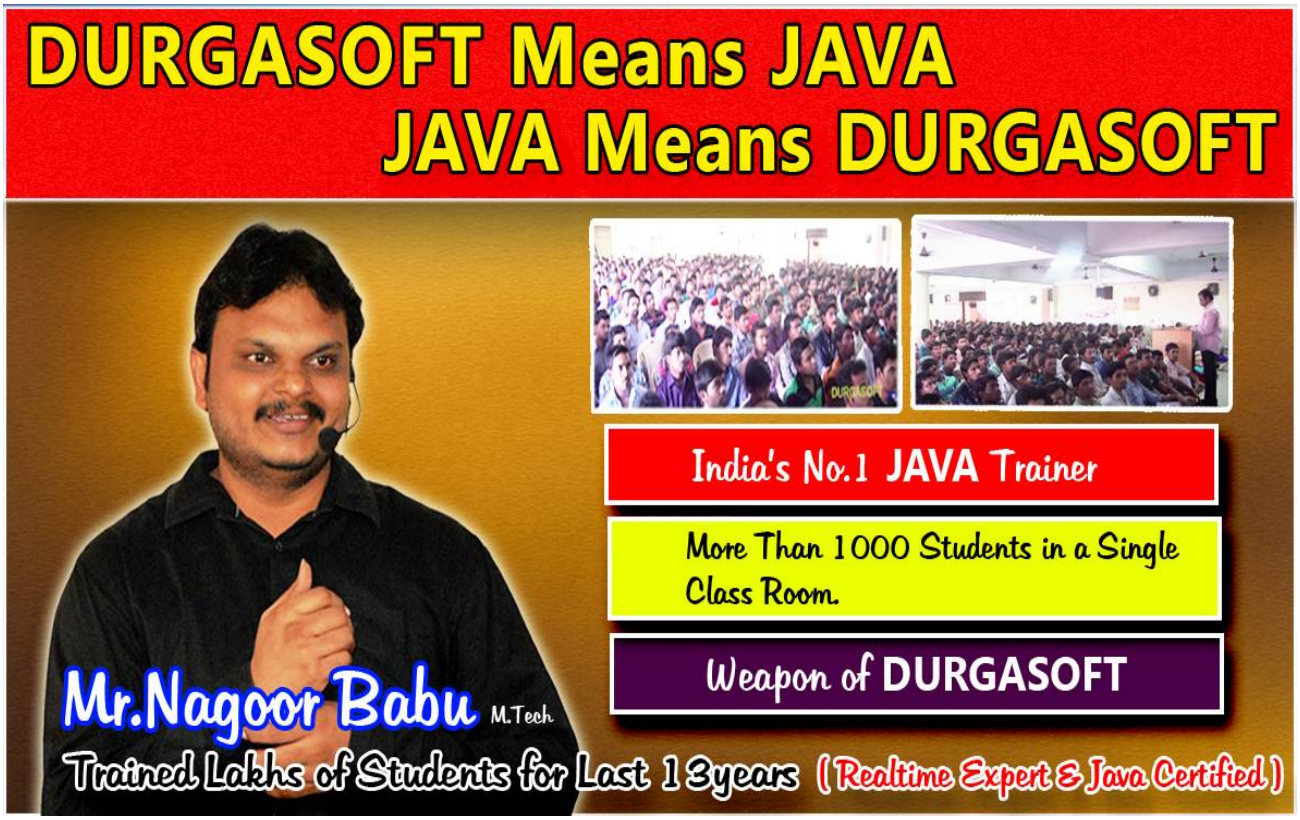
${fn.toUpperCase (a)}

${fn.contains(a, "Software")}

${fn.startsWith(a, "Durga")}

${fn.endsWith(a, "Solutions")}

${fn.substring(a, 4, 20)}


DURGASOFT Means JAVA
JAVA Means DURGASOFT
India's No.1 JAVA Trainer
More Than 1000 Students in a Single Class Room.
Weapon of DURGASOFT
Mr.Nagoor Babu M.Tech
Trained Lakhs of Students for Last 13years (Realtime Expert & Java Certified)