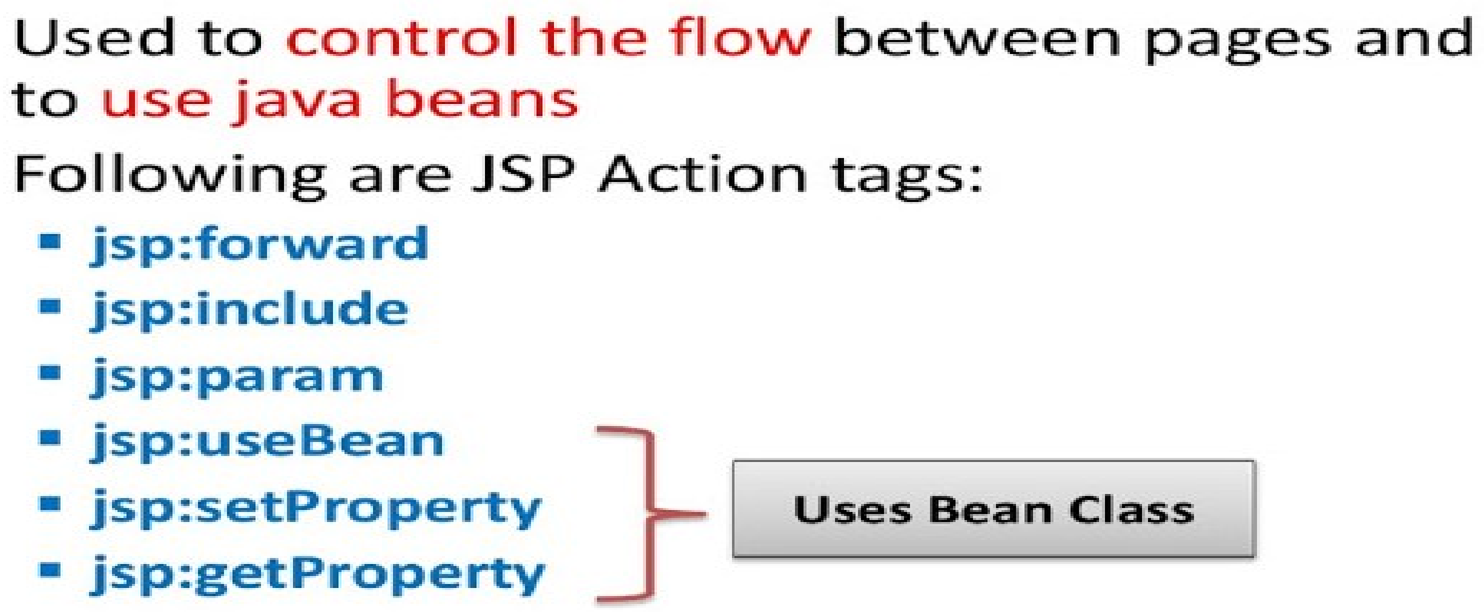
JSP-Action Tags & Custom Tags

Presented by

Sagar
Java Consultant

Action Tags

- Action tags effects the runtime behavior of JSP
- The response of the action will be sent to the client



Used to control the flow between pages and to use java beans

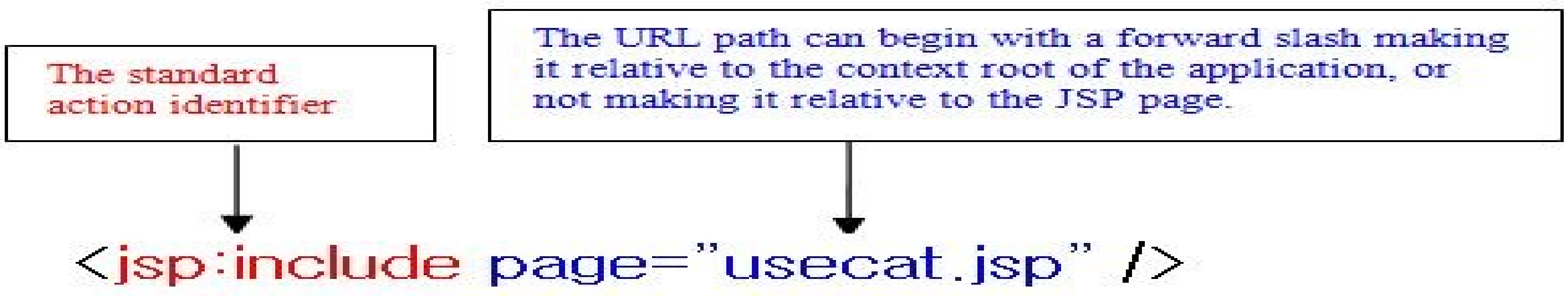
Following are JSP Action tags:

- jsp:forward
- jsp:include
- jsp:param
- jsp:useBean
- jsp:setProperty
- jsp:getProperty

Uses Bean Class

<jsp:include>

- Includes another web resource (html/jsp/servlet)



The standard action identifier

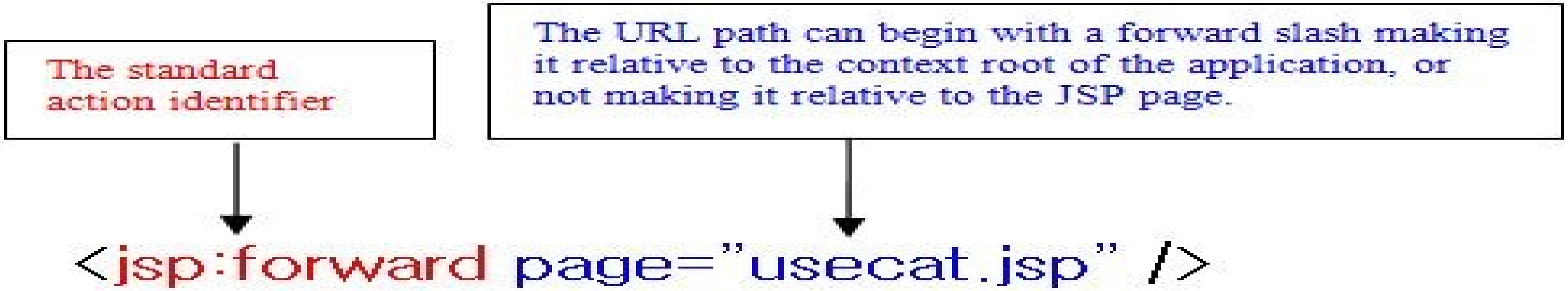
The URL path can begin with a forward slash making it relative to the context root of the application, or not making it relative to the JSP page.

```
<jsp:include page="usecat.jsp" />
```

**Example

<jsp:forward>

Forwards to another web resource (html/jsp/servlet)



The standard action identifier

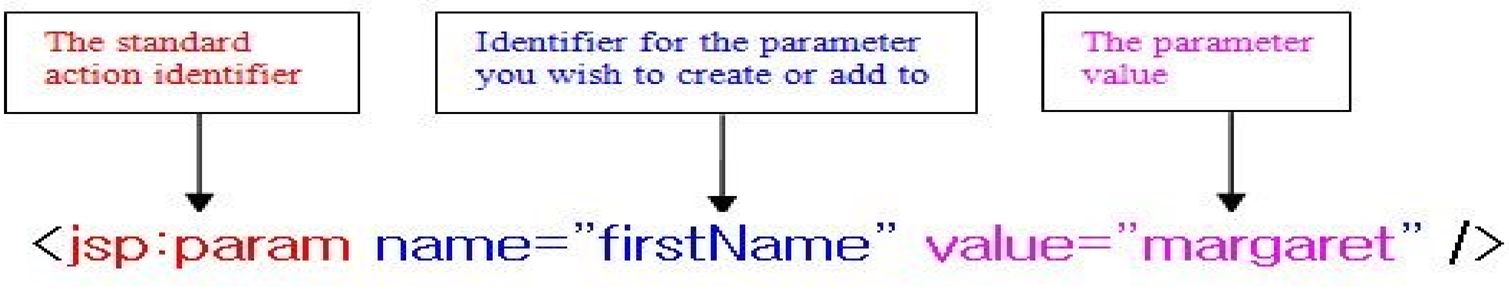
The URL path can begin with a forward slash making it relative to the context root of the application, or not making it relative to the JSP page.

```
<jsp:forward page=”usecat.jsp” />
```

**Example

<jsp:param>

The **<jsp:param>** action is used to provide information in the form of key/value pair



The standard action identifier

Identifier for the parameter you wish to create or add to

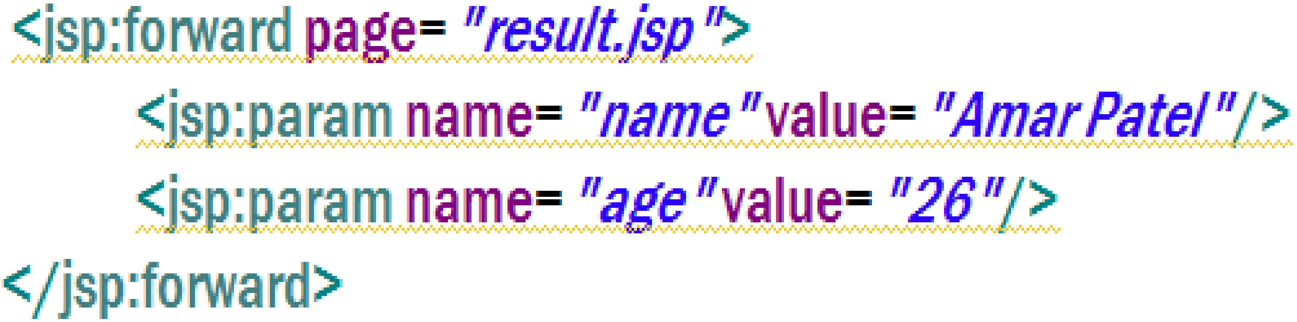
The parameter value

```
<jsp:param name="firstName" value="margaret" />
```

**Example

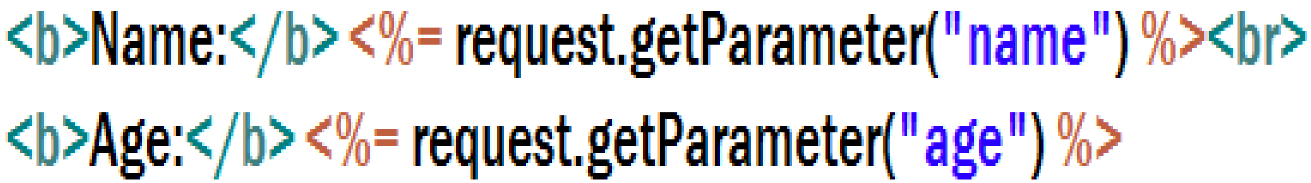
<jsp:param> example

index.jsp



```
<jsp:forward page="result.jsp">
        <jsp:param name="name" value="Amar Patel"/>
        <jsp:param name="age" value="26"/>
</jsp:forward>
```

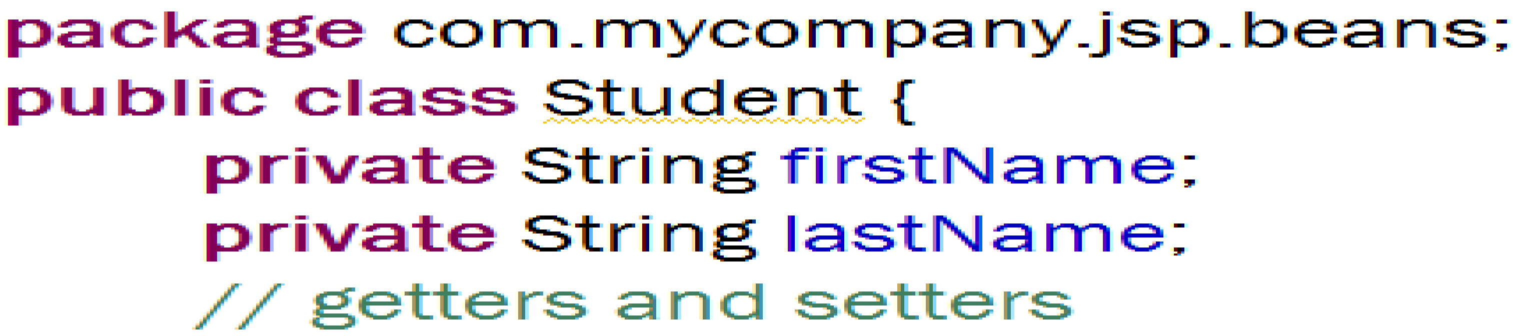
result.jsp



```
<b>Name:</b> <%= request.getParameter("name") %><br>
<b>Age:</b> <%= request.getParameter("age") %>
```

<jsp:useBean>

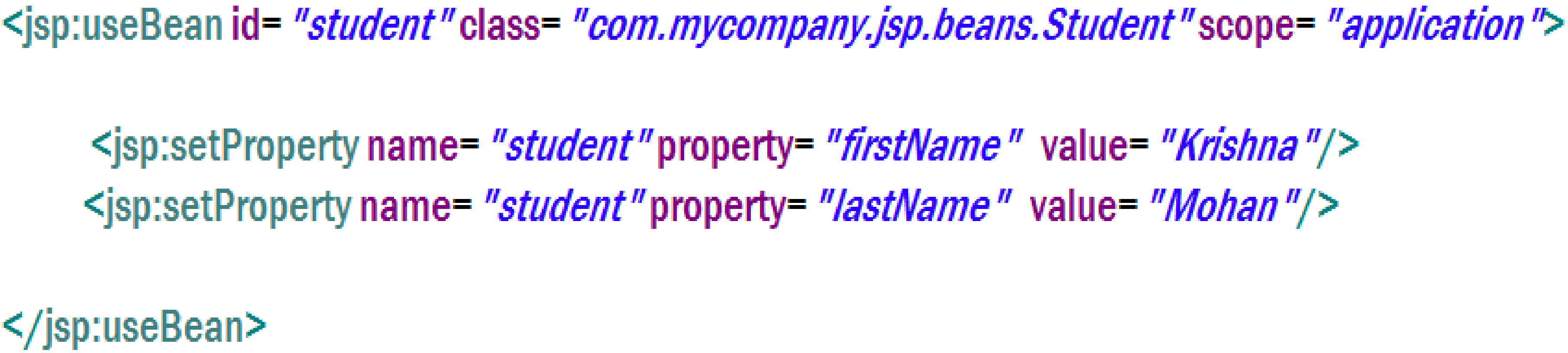
- Using a POJO / Bean in JSP
- Properties of Bean are used in JSP to set and to get
- Bean can be used through out the application
- Example:



```
package com.mycompany.jsp.beans;
public class Student {
        private String firstName;
        private String lastName;
        // getters and setters
```

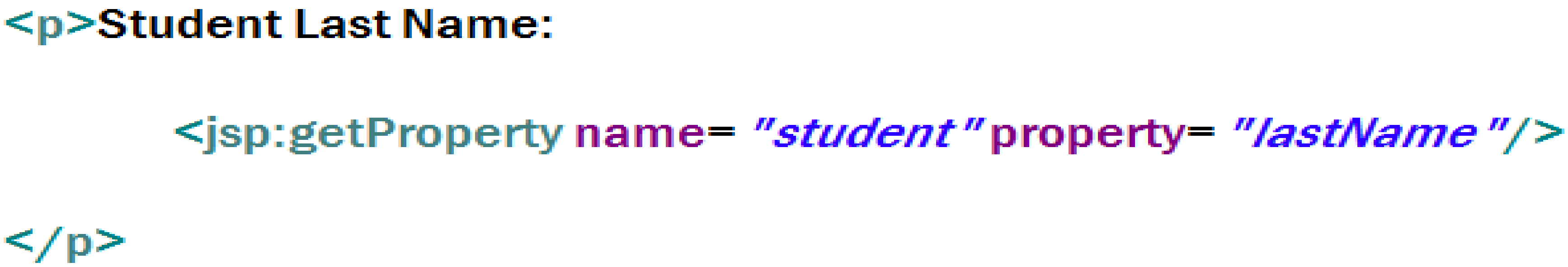
<jsp:setProperty…> & <jsp:getProperty…>

index.jsp



```
<jsp:useBean id= "student" class= "com.mycompany.jsp.beans.Student" scope= "application">

    <jsp:setProperty name= "student" property= "firstName"  value= "Krishna"/>
    <jsp:setProperty name= "student" property= "lastName"  value= "Mohan"/>

</jsp:useBean>
```

result.jsp



```
<p>Student Last Name:

        <jsp:getProperty name= "student" property= "lastName"/>

</p>
```

JSP Custom Tags

Custom tags are user-defined elements that extend the functionality of JSP pages.

Key Concepts :

- Tag Library: A collection of related custom tags.
- Tag Handler: The Java class that implements the logic for a custom tag.
- Tag Attributes: Parameters passed to the custom tag.

Benefits of Using Custom Tags :

- Encapsulation: Hide complex logic within the tag.
- Reusability: Use the same tag multiple times.
- Maintainability: Improve code organization and readability.
- Separation of Concerns: Clearly separate presentation and business logic.

Custom Tags Creation

1. Create a Tag Handler Class:

    - Extend SimpleTagSupport for simple tags or TagSupport for complex tags.
    - Implement the required methods (e.g., doTag, setAttributes).
    - Write the logic for the tag's behavior.
    - Create a Tag Library Descriptor (TLD)

2. Define the tag name, attributes, and other metadata.

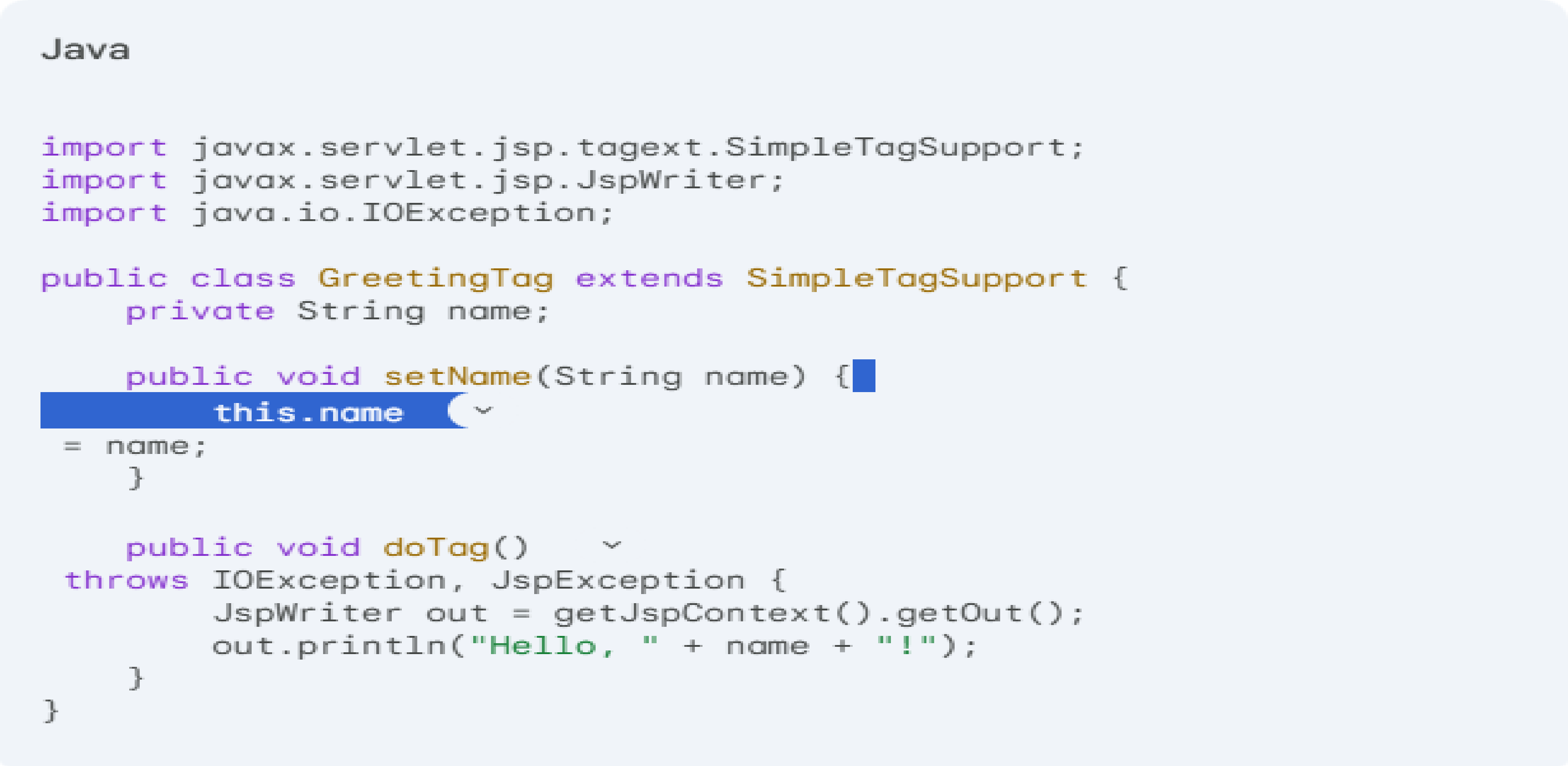
    - Use the taglib directive in JSP pages to import the tag library.
    - Use the Custom Tag in JSP:

3. Use the tag prefix defined in the TLD.

    - Specify attributes as needed.

Creating Custom Tags - Example

1. Tag Handler:

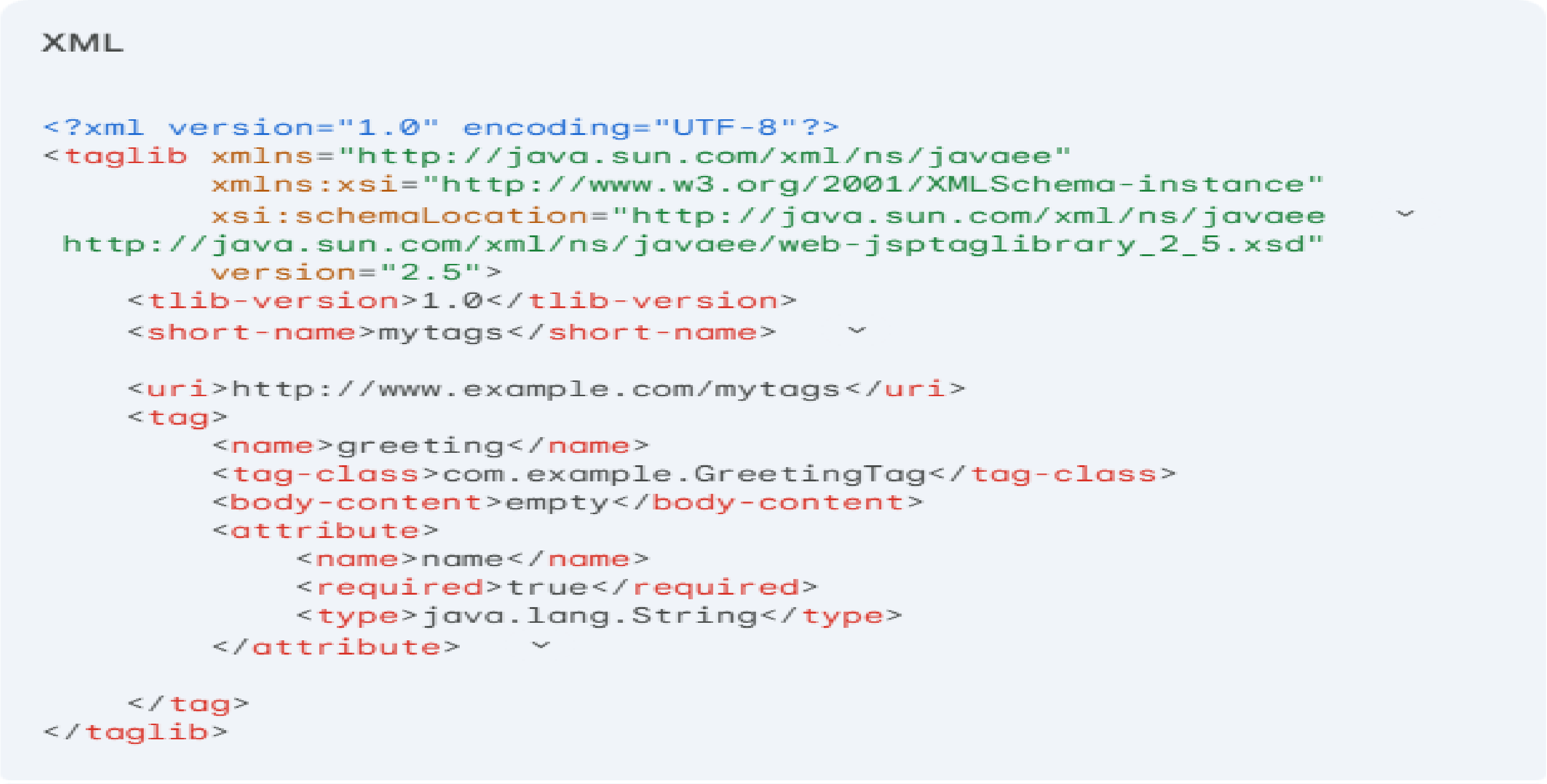


Java

```
import javax.servlet.jsp.tagext.SimpleTagSupport;
import javax.servlet.jsp.JspWriter;
import java.io.IOException;

public class GreetingTag extends SimpleTagSupport {
    private String name;

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

    public void doTag()
 throws IOException, JspException {
        JspWriter out = getJspContext().getOut();
        out.println("Hello, " + name + "!");
    }
}
```

Creating Custom Tags …

2. Creating Tag Library Descriptor (TLD):

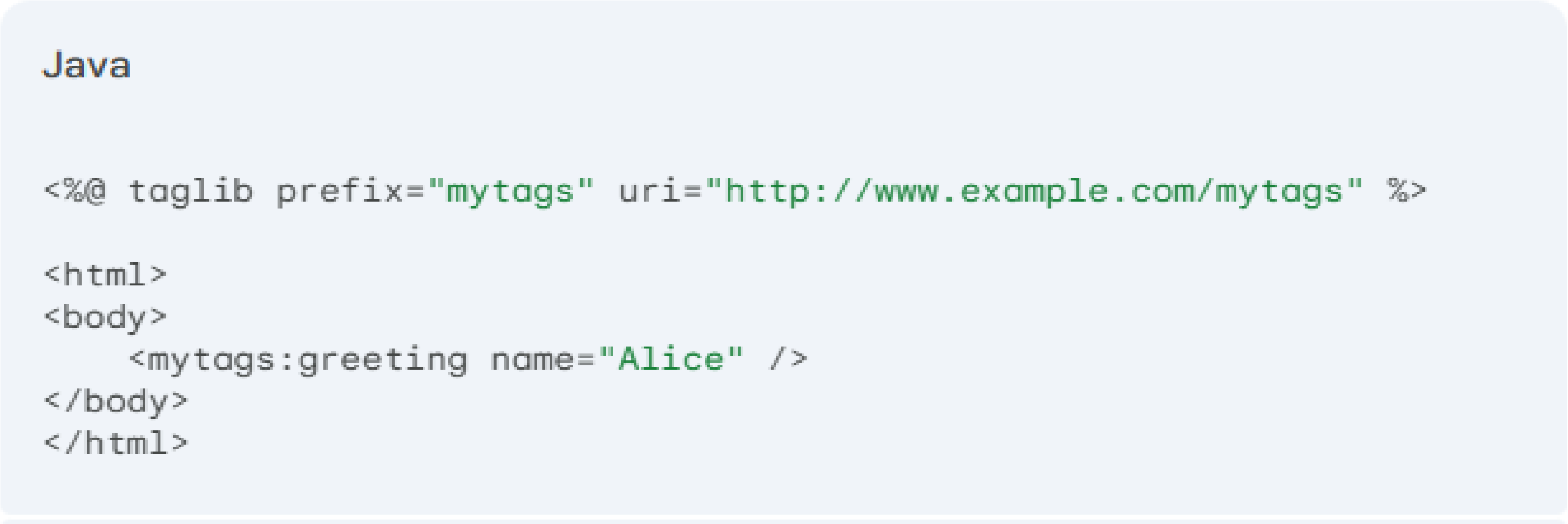


XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<taglib xmlns="http://java.sun.com/xml/ns/javaee"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
 http://java.sun.com/xml/ns/javaee/web-jsptaglibrary_2_5.xsd"
        version="2.5">
    <tlib-version>1.0</tlib-version>
    <short-name>mytags</short-name>

    <uri>http://www.example.com/mytags</uri>
    <tag>
        <name>greeting</name>
        <tag-class>com.example.GreetingTag</tag-class>
        <body-content>empty</body-content>
        <attribute>
            <name>name</name>
            <required>true</required>
            <type>java.lang.String</type>
        </attribute>

    </tag>
</taglib>
```

Creating Custom Tags …

3. Using the custom tag in JSP

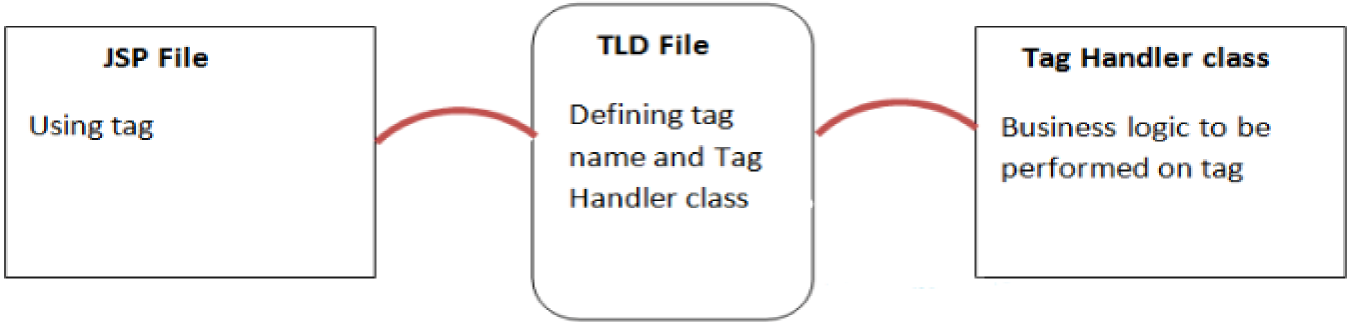


Java

```
<%@ taglib prefix="mytags" uri="http://www.example.com/mytags" %>

<html>
<body>
    <mytags:greeting name="Alice" />
</body>
</html>
```

**Demo

Flow of Custom Tags



JSP File
Using tag
TLD File
Defining tag name and Tag Handler class
Tag Handler class
Business logic to be performed on tag

**Demo

Thank You