JAVA BEAN

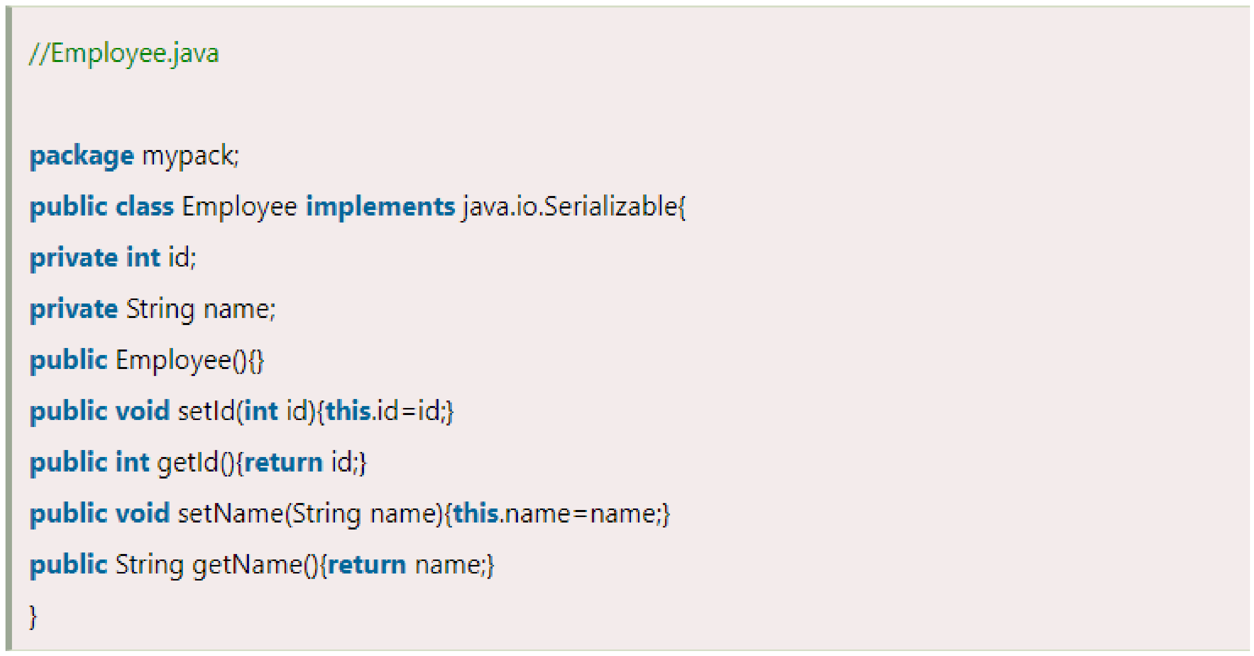
Java Beans are classes that encapsulates many objects into a single object (the bean). A JavaBean is a Java class that should follow the following conventions:

- It should have a no-arg constructor.
- It should be Serializable.
- It should provide methods to set and get the values of the properties, known as getter and setter methods.

Why use JavaBean?

According to Java white paper, it is a reusable software component. A bean encapsulates many objects into one object so that we can access this object from multiple places. Moreover, it provides easy maintenance..

Simple example of JavaBean class



```java
//Employee.java

package mypack;
public class Employee implements java.io.Serializable{
private int id;
private String name;
public Employee(){}
public void setId(int id){this.id=id;}
public int getId(){return id;}
public void setName(String name){this.name=name;}
public String getName(){return name;}
}
```

How to access the JavaBean class?

To access the JavaBean class, we should use getter and setter methods.

```
package mypack;
public class Test{
public static void main(String args[]){
Employee e=new Employee();//object is created
e.setName("Arjun");//setting value to the object
System.out.println(e.getName());
}}
```

Note: There are two ways to provide values to the object. One way is by constructor and second is by setter method.

JavaBean Properties

A JavaBean property is a named feature that can be accessed by the user of the object. The feature can be of any Java data type, containing the classes that you define.

A JavaBean property may be read, write, read-only, or write-only. JavaBean features are accessed through two methods in the JavaBean's implementation class:\

1. getPropertyName ()
For example, if the property name is first Name, the method name would be getFirstName() to read that Property. This method is called the accessory.

2. setPropertyName ()
For example, if the property name is firstName, the method name would be setFirstName() to write that property. This method is called the mutator.

Advantages Of JavaBean

- The properties, events, and methods of a bean can be exposed to another application.
- A bean may register to receive events from other objects and can generate events that are sent to those other objects.
- Auxiliary software can be provided to help configure a bean.
- The configuration settings of a bean can be saved to persistent storage and restored.

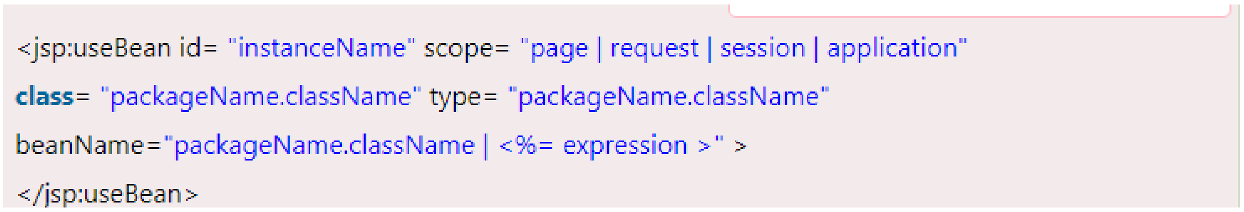
Disadvantages of JavaBean

- A class with a zero-argument constructor is subject to being instantiated in an invalid state.[1] If such a class is instantiated manually by a developer (rather than automatically by some kind of framework), the developer might not realize that the class has been improperly instantiated. The compiler cannot detect such a problem, and even if it is documented, there is no guarantee that the developer will see the documentation.
- JavaBeans are inherently mutable and so lack the advantages offered by immutable objects.[1]
- Having to create getters for every property and setters for many, most, or all of them can lead to an immense quantity of boilerplate code. This can be mitigated using tools like Lombok.

JSP:useBean action tag

The jsp:useBean action tag is used to locate or instantiate a bean class. If bean object of the Bean class is already created, it doesn't create the bean depending on the scope. But if object of bean is not created, it instantiates the bean.

Syntax of jsp:useBean action tag



```
<jsp:useBean id= "instanceName" scope= "page | request | session | application"
class= "packageName.className" type= "packageName.className"
beanName="packageName.className | <%= expression >" >
</jsp:useBean>
```

Attributes and Usage of jsp:useBean action tag

1. **id:** is used to identify the bean in the specified scope.
2. **scope:** represents the scope of the bean. It may be page, request, session or application. The default scope is page.
   - **page:** specifies that you can use this bean within the JSP page. The default scope is page.
   - **request:** specifies that you can use this bean from any JSP page that processes the same request. It has wider scope than page.
   - **session:** specifies that you can use this bean from any JSP page in the same session whether processes the same request or not. It has wider scope than request.
   - **application:** specifies that you can use this bean from any JSP page in the same application. It has wider scope than session.

3. **class:** instantiates the specified bean class (i.e. creates an object of the bean class) but it must have no-arg or no constructor and must not be abstract.
4. **type:** provides the bean a data type if the bean already exists in the scope. It is mainly used with class or beanName attribute. If you use it without class or beanName, no bean is instantiated.
5. **beanName:** instantiates the bean using the java.beans.Beans.instantiate() method.

Simple example of jsp:useBean action tag

In this example, we are simply invoking the method of the Bean class.

For the example of setProperty, getProperty and useBean tags, visit next page.

Calculator.java (a simple Bean class)

```java
package com.javatpoint;
public class Calculator{

public int cube(int n){return n*n*n;}

}
```

index.jsp file

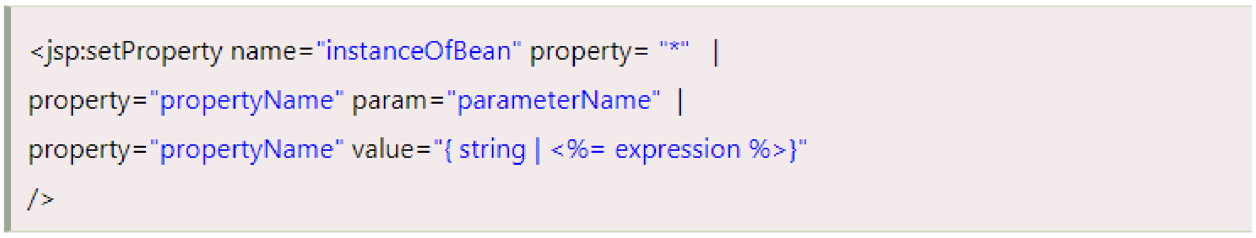
```jsp
<jsp:useBean id="obj" class="com.javatpoint.Calculator"/>

<%
int m=obj.cube(5);
out.print("cube of 5 is "+m);
%>
```

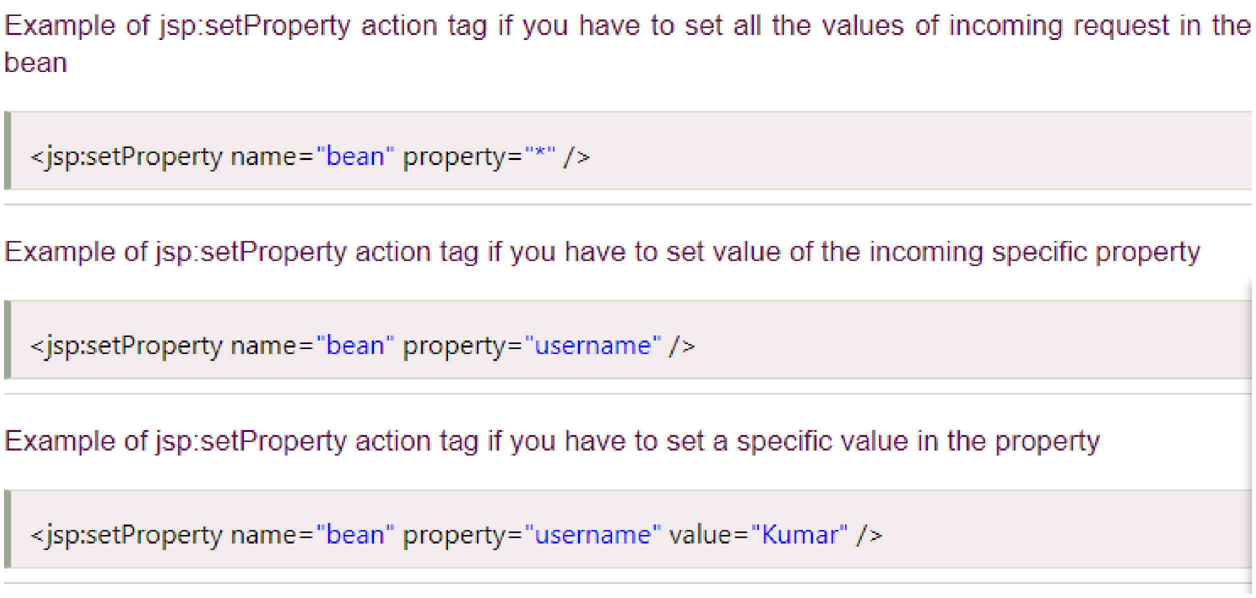
JSP:setProperty and JSP:getProperty action tags

The setProperty and getProperty action tags are used for developing web application with Java Bean. In web devlopment, bean class is mostly used because it is a reusable software component that represents data.
The jsp:setProperty action tag sets a property value or values in a bean using the setter method.

Syntax of jsp:setProperty action tag



```
<jsp:setProperty name="instanceOfBean" property= "*"   |
property="propertyName" param="parameterName"  |
property="propertyName" value="{ string | <%= expression %>}"
/>
```



Example of jsp:setProperty action tag if you have to set all the values of incoming request in the bean

```
<jsp:setProperty name="bean" property="*" />
```

Example of jsp:setProperty action tag if you have to set value of the incoming specific property

```
<jsp:setProperty name="bean" property="username" />
```

Example of jsp:setProperty action tag if you have to set a specific value in the property

```
<jsp:setProperty name="bean" property="username" value="Kumar" />
```