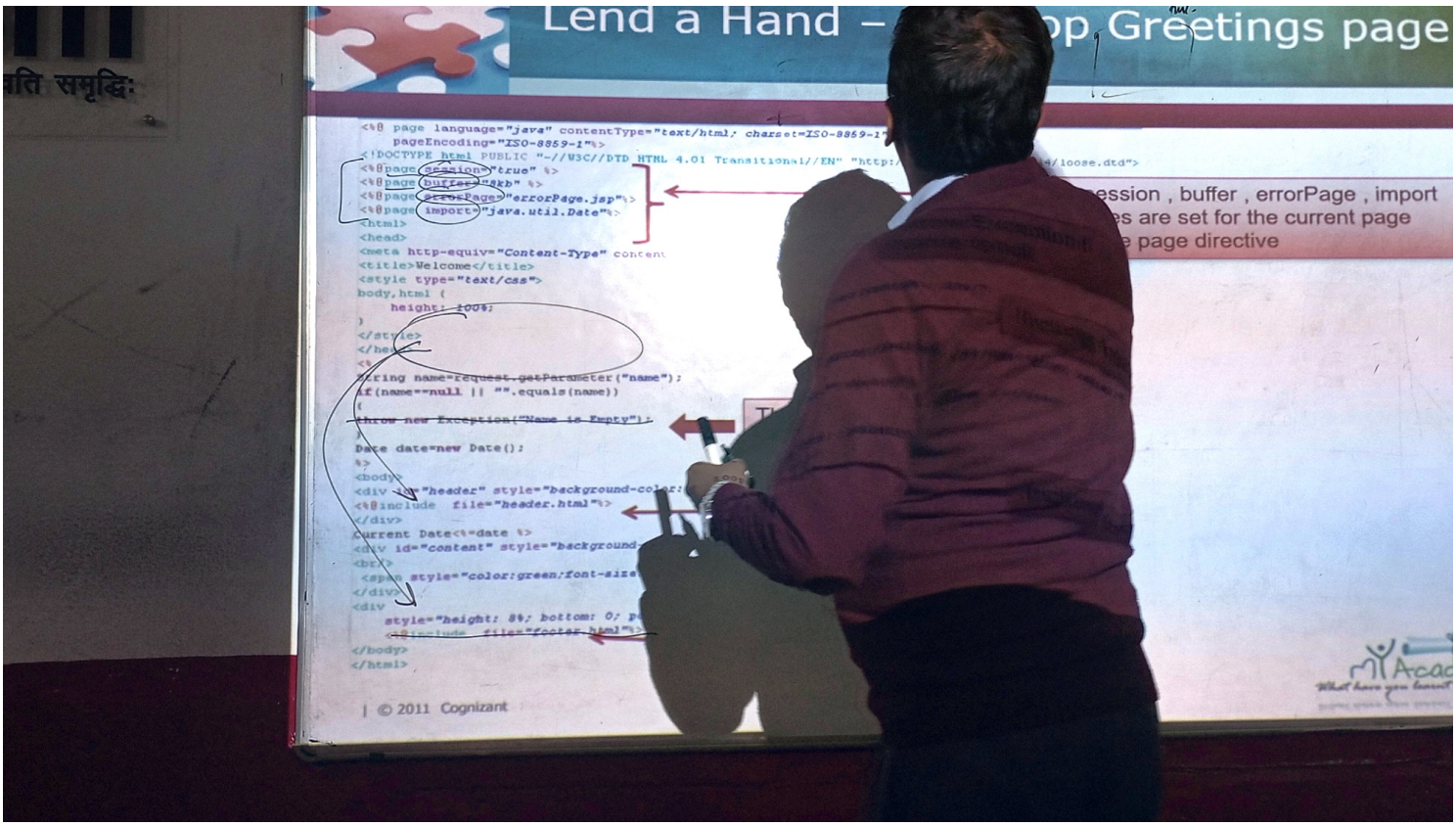
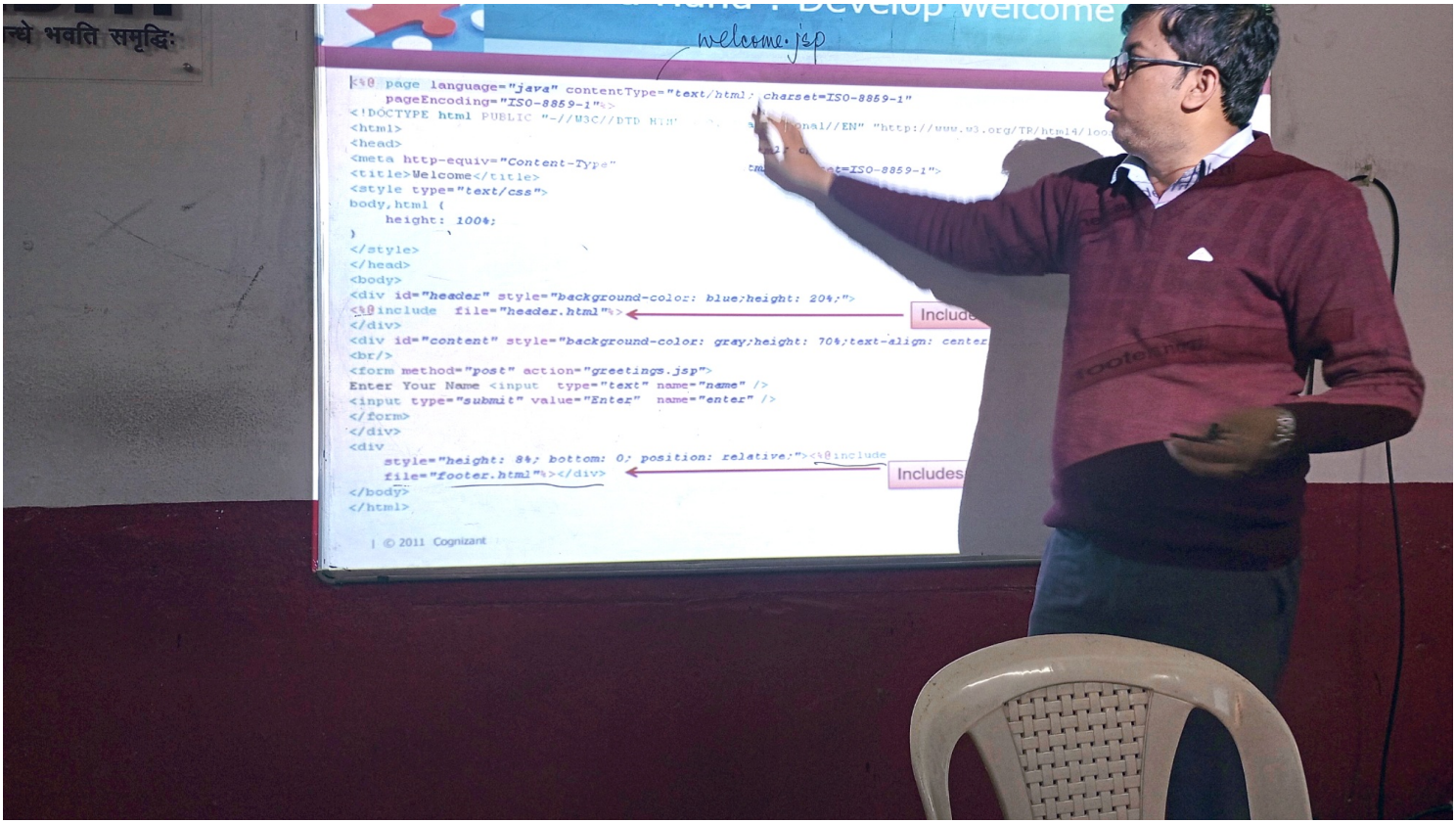

Lend a Hand –
op Greetings page
ession , buffer , errorPage , import
es are set for the current page
e page directive
© 2011 Cognizant

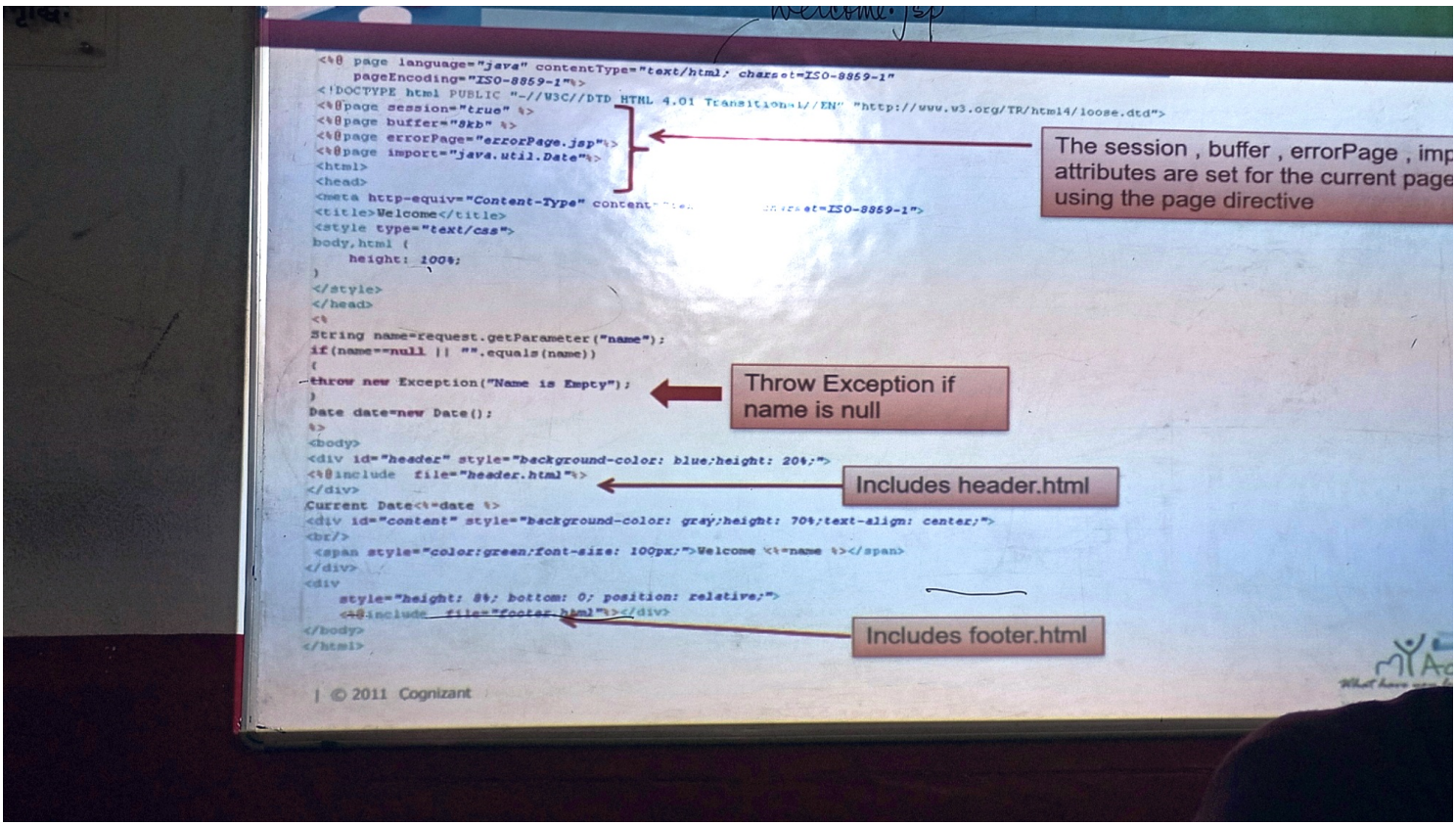

welcome.jsp
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTM
<html>
<head>
<meta http-equiv="Content-Type"
<title>Welcome</title>
<style type="text/css">
body,html {
height: 100%;
}
</style>
</head>
<body>

<%@include file="header.html"%>

<div id="content" style="background-color: gray;height: 70%;text-align: center

<form method="post" action="greetings.jsp">
Enter Your Name <input type="text" name="name" />
<input type="submit" value="Enter" name="enter" />
</form>

<div
style="height: 8%; bottom: 0; position: relative;"><%@include
file="footer.html"%>
</body>
</html>
© 2011 Cognizant
Include
Includes

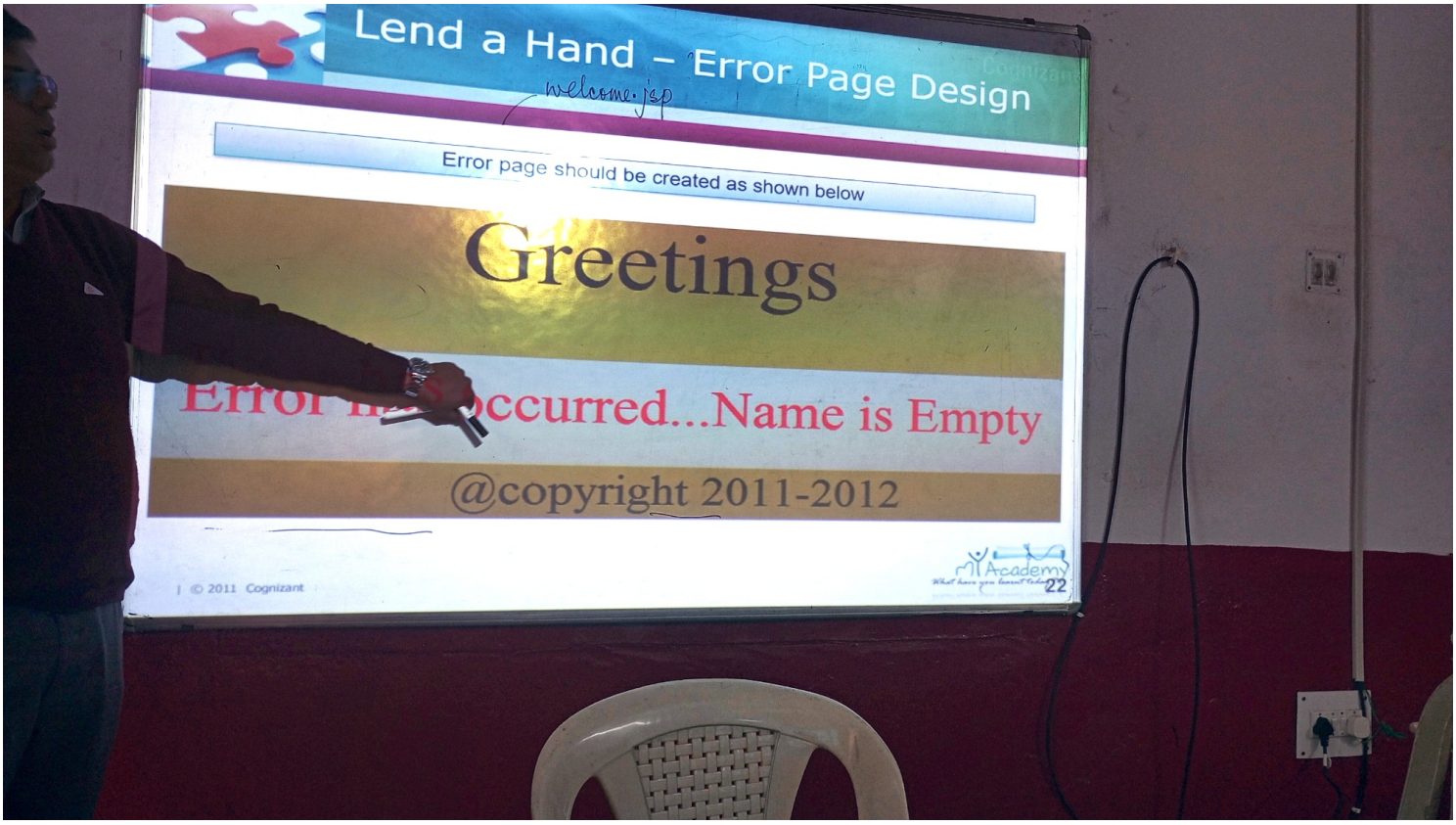

welcome.jsp
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<%@page session="true" %>
<%@page buffer="8kb" %>
<%@page errorPage="errorPage.jsp"%>
<%@page import="java.util.Date"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Welcome</title>
<style type="text/css">
body,html {
height: 100%;
}
</style>
</head>
<%
String name=request.getParameter("name");
if(name==null || "".equals(name))
{
throw new Exception("Name is Empty");
}
Date date=new Date();
%>
<body>

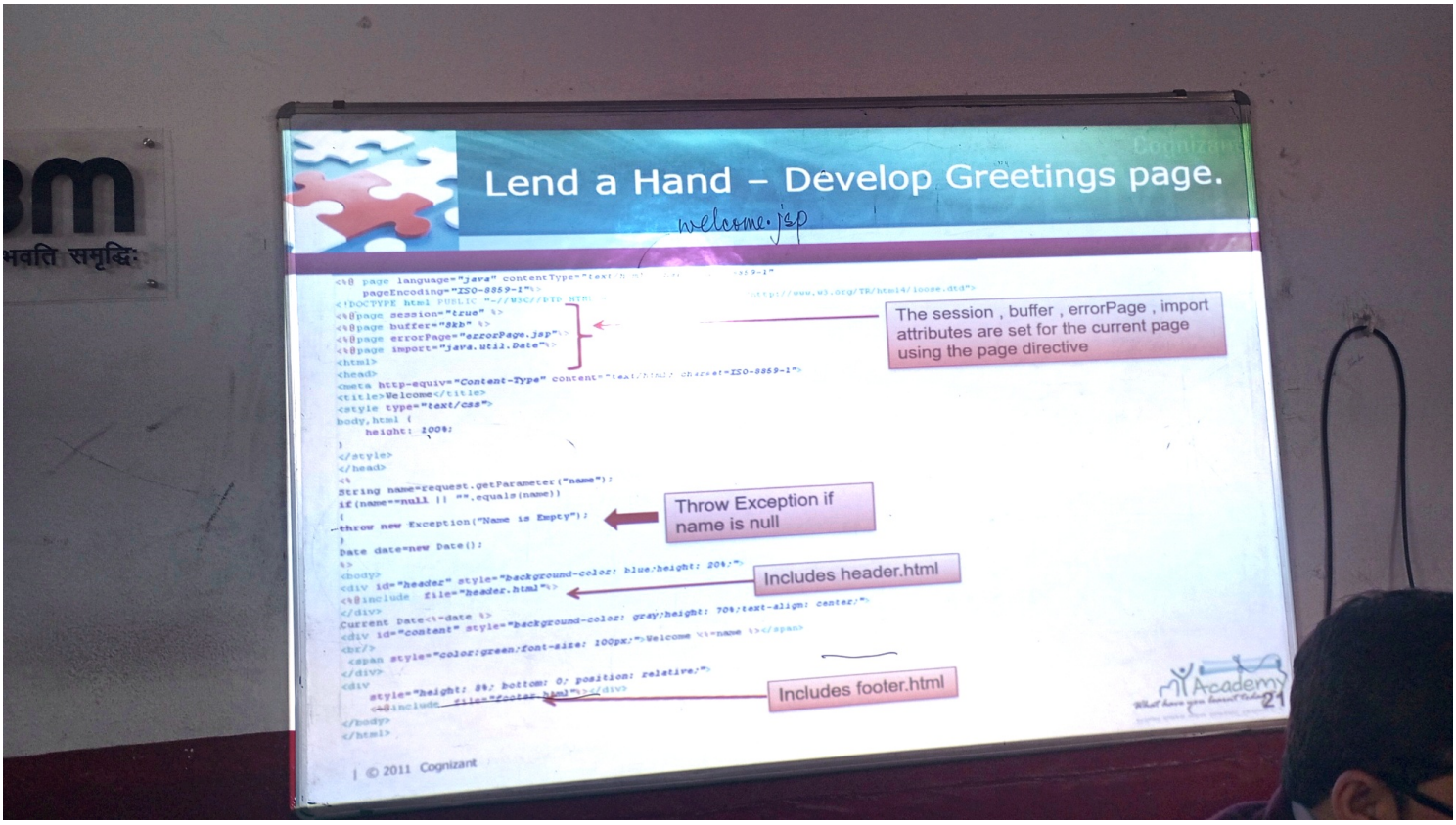
<%@include file="header.html"%>

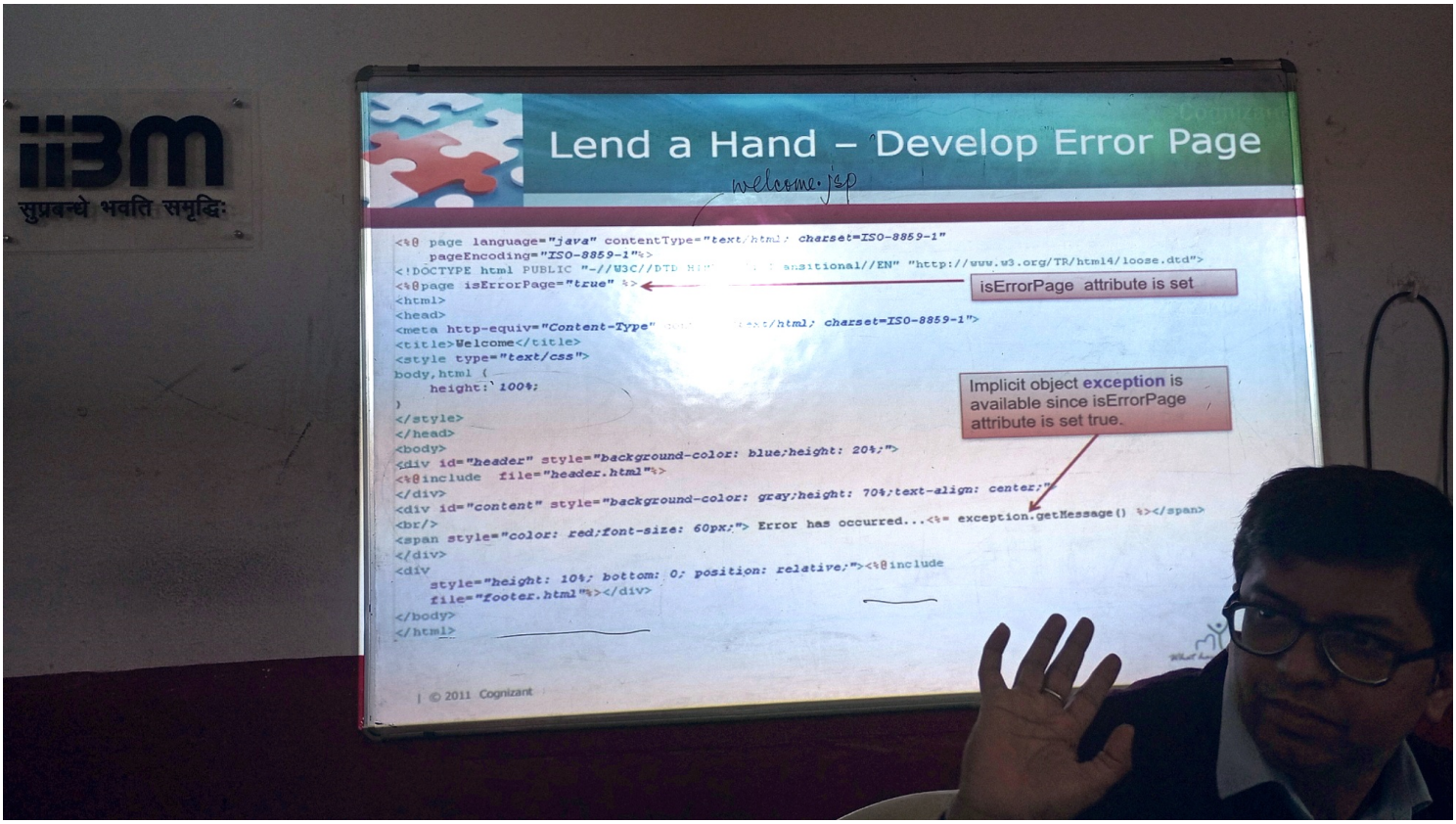
Current Date<%=date %>

Welcome <%=name %>

<div
style="height: 8%; bottom: 0; position: relative;">
<%@include file="footer.html"%>
</body>
</html>
The session , buffer , errorPage , imp
attributes are set for the current page
using the page directive
Throw Exception if
name is null
Includes header.html
Includes footer.html
© 2011 Cognizant


Lend a Hand – Error Page Design
welcome.jsp
Error page should be created as shown below
Greetings
ccurred...Name is Empty
@copyright 2011-2012
© 2011 Cognizant
22

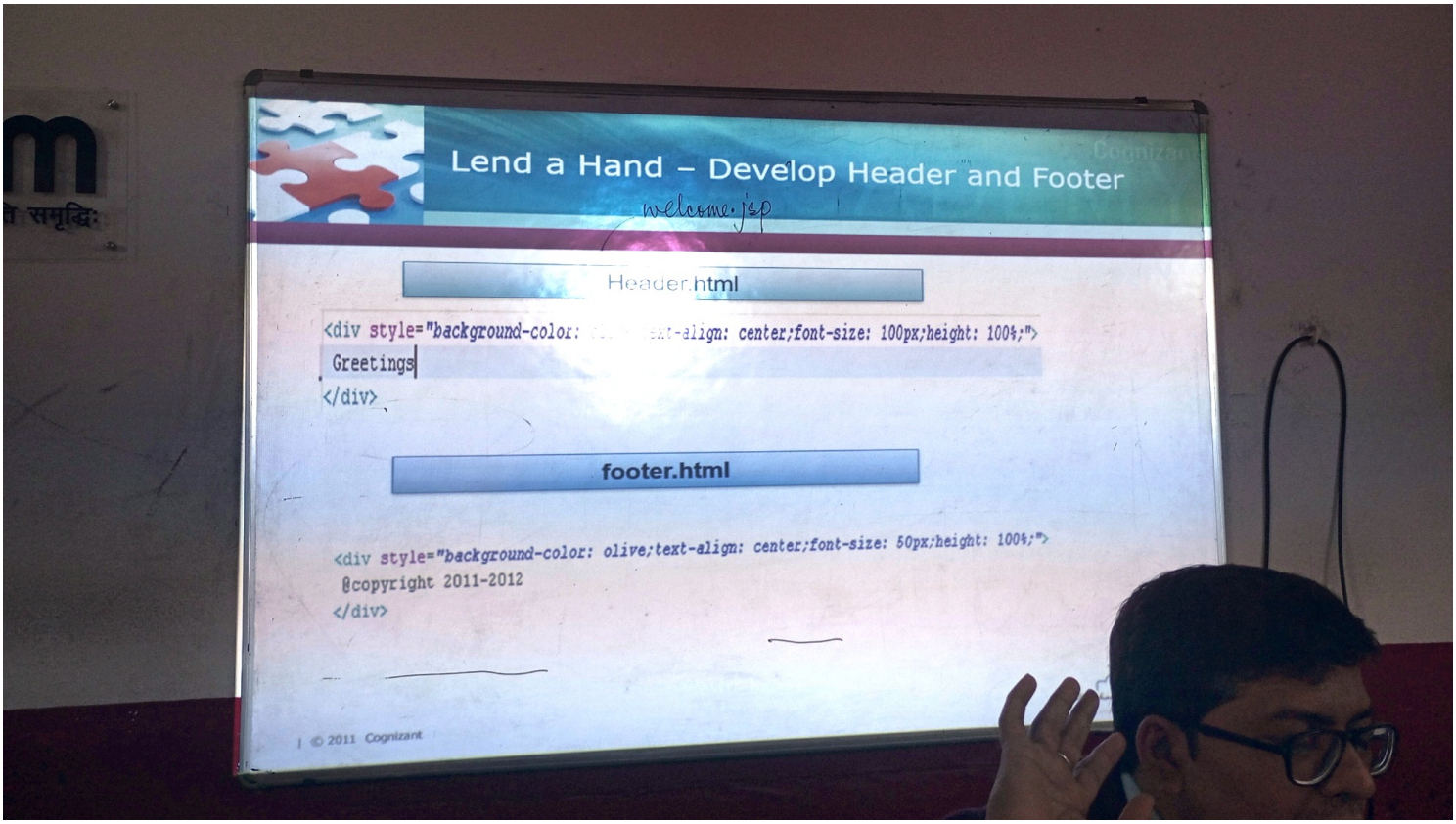

Lend a Hand – Develop Greetings page.
welcome.jsp
The session , buffer , errorPage , import attributes are set for the current page using the page directive
Throw Exception if name is null
Includes header.html
Includes footer.html
© 2011 Cognizant


Lend a Hand – Develop Error Page
welcome.jsp
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<%@page isErrorPage="true" %>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Welcome</title>
<style type="text/css">
body,html {
height: 100%;
}
</style>
</head>
<body>

<%@include file="header.html"%>

 Error has occurred...<%= exception.getMessage() %>

<div
style="height: 10%; bottom: 0; position: relative;"><%@include
file="footer.html"%>
</body>
</html>
isErrorPage attribute is set
Implicit object exception is available since isErrorPage attribute is set true.
© 2011 Cognizant

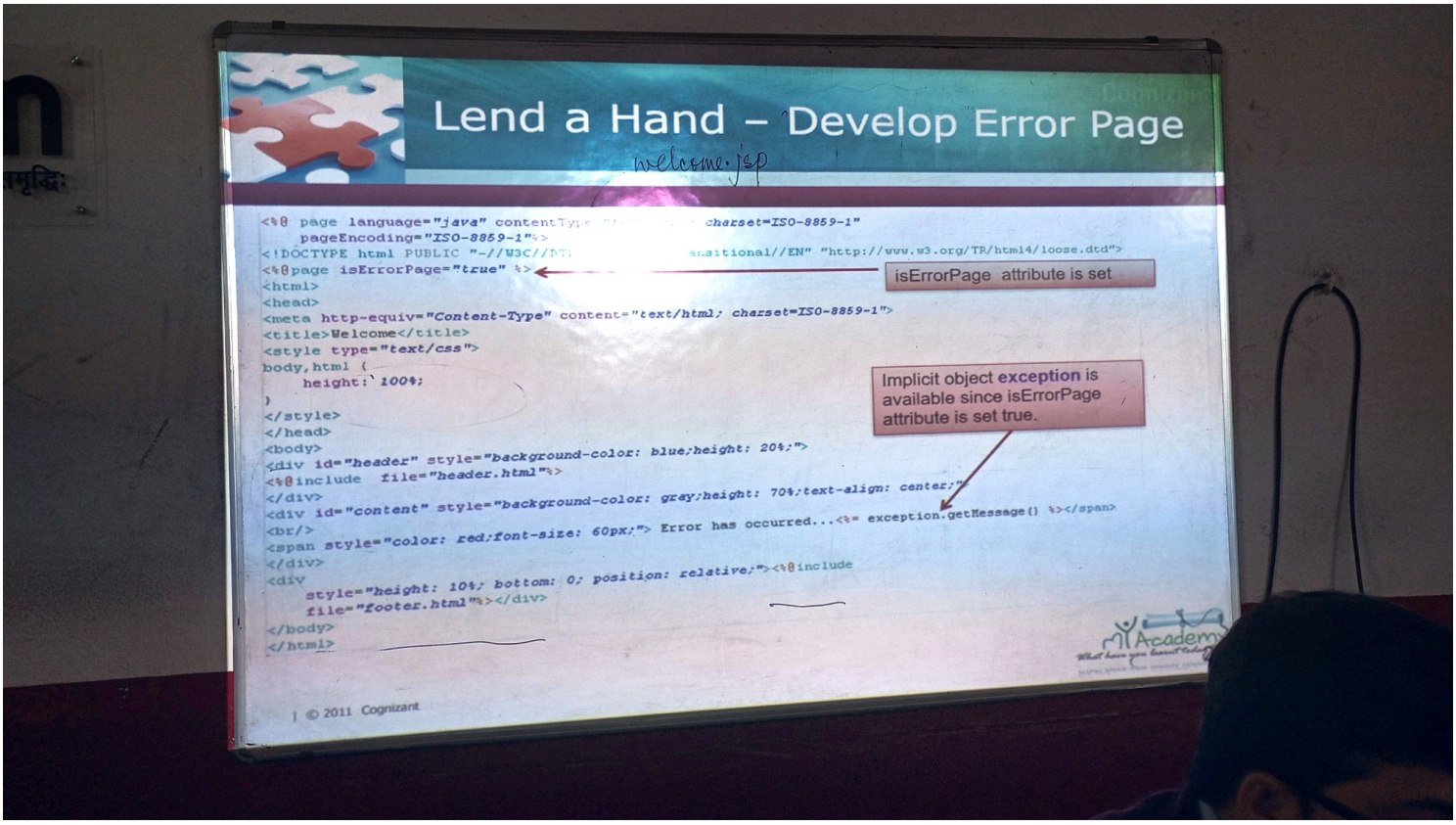

Lend a Hand – Develop Header and Footer
welcome.jsp
Header.html

Greetings

footer.html

@copyright 2011-2012

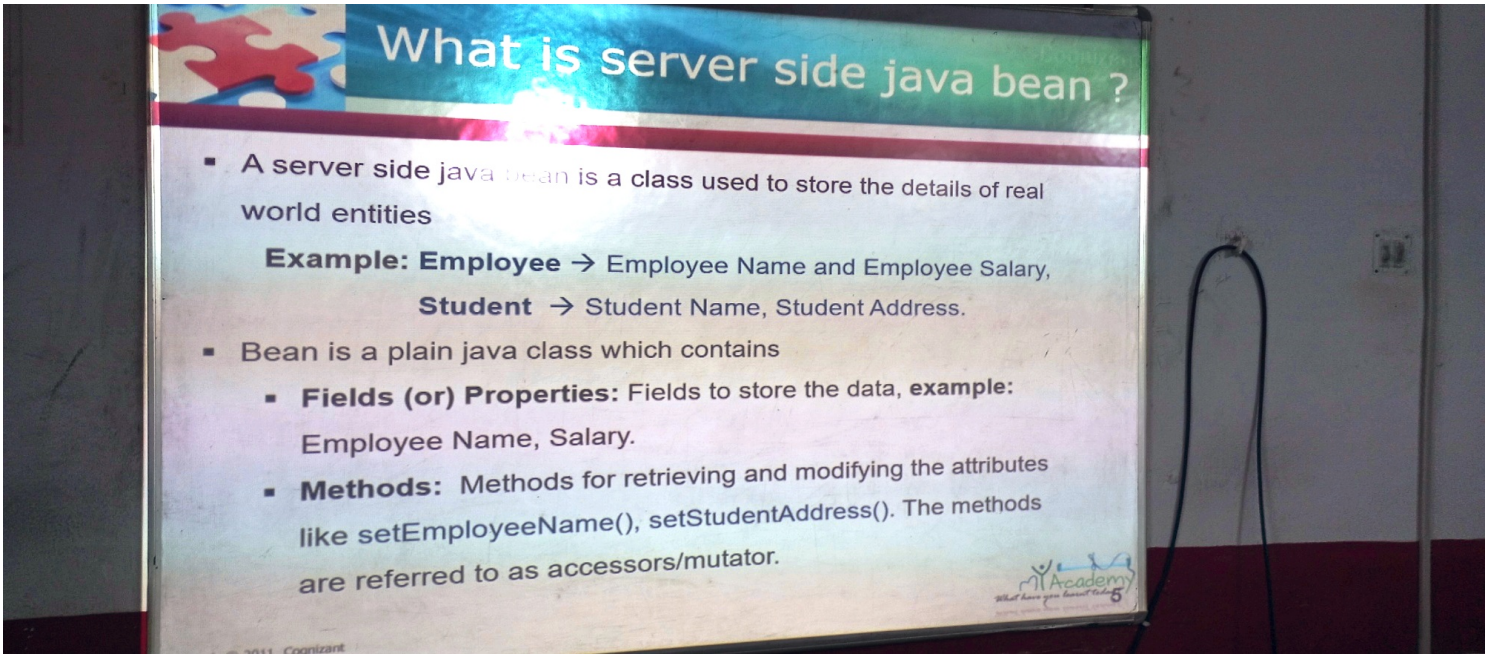
© 2011 Cognizant

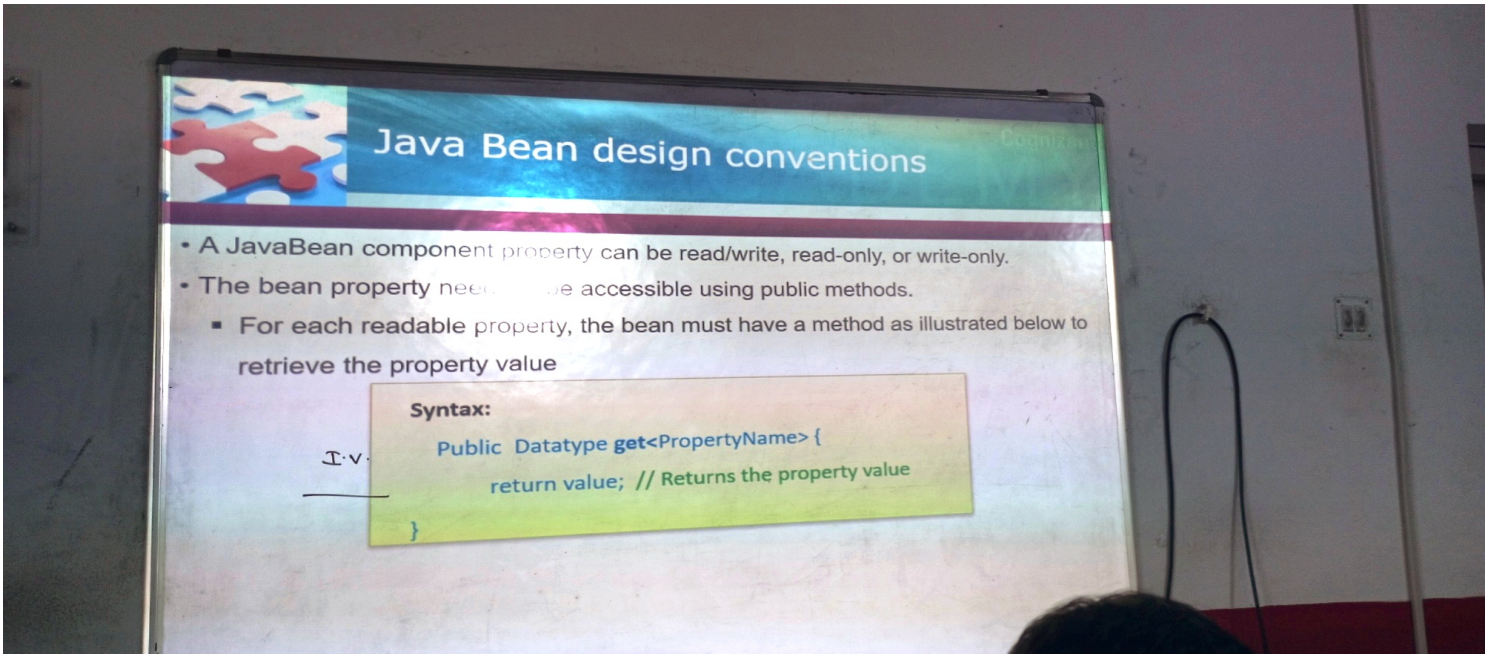

Lend a Hand – Develop Error Page
welcome.jsp
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<%@page isErrorPage="true" %>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Welcome</title>
<style type="text/css">
body,html {
height: 100%;
}
</style>
</head>
<body>

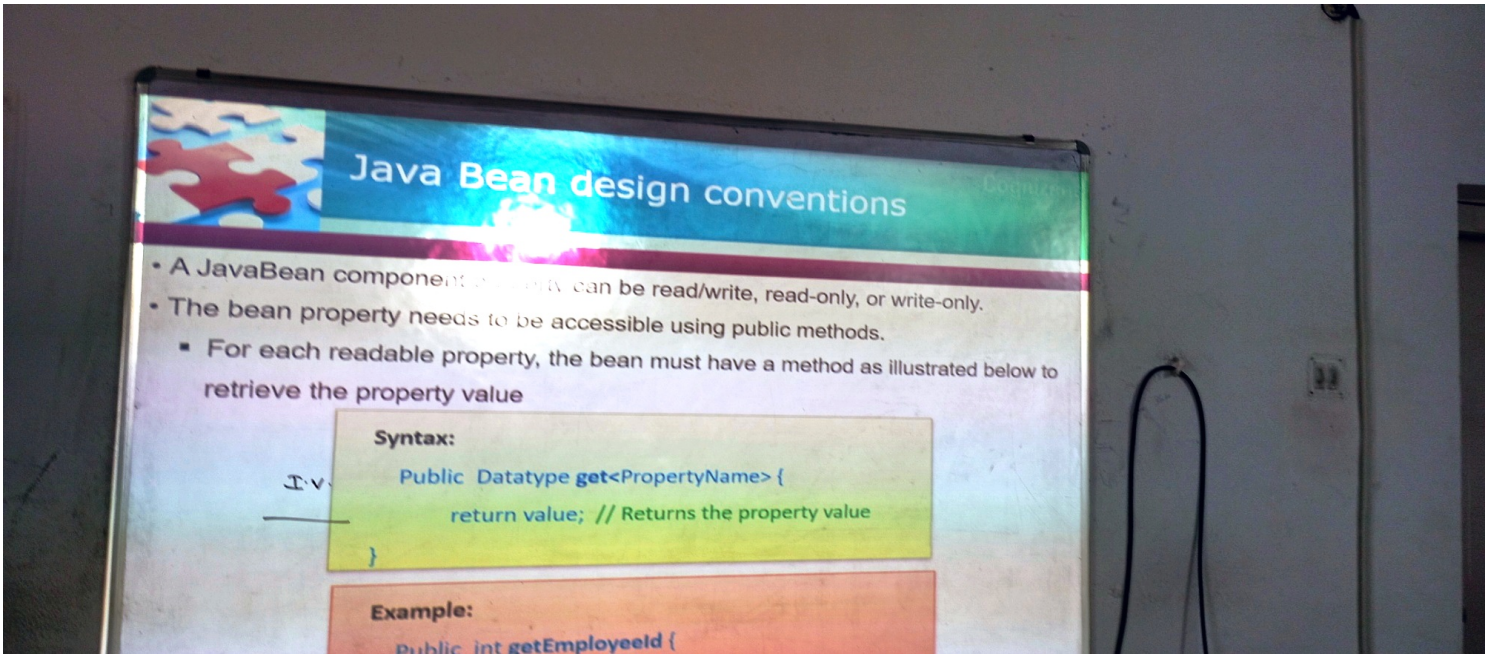
<%@include file="header.html"%>

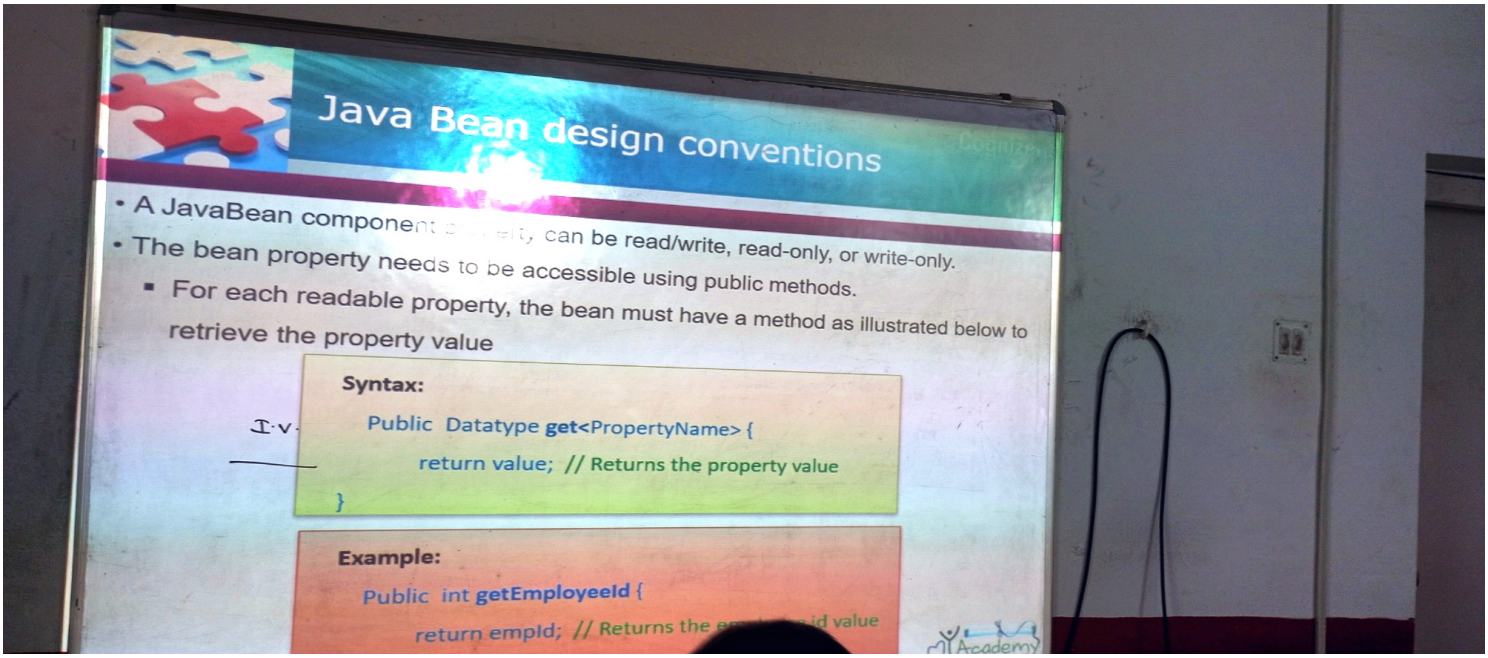
 Error has occurred...<%= exception.getMessage() %>

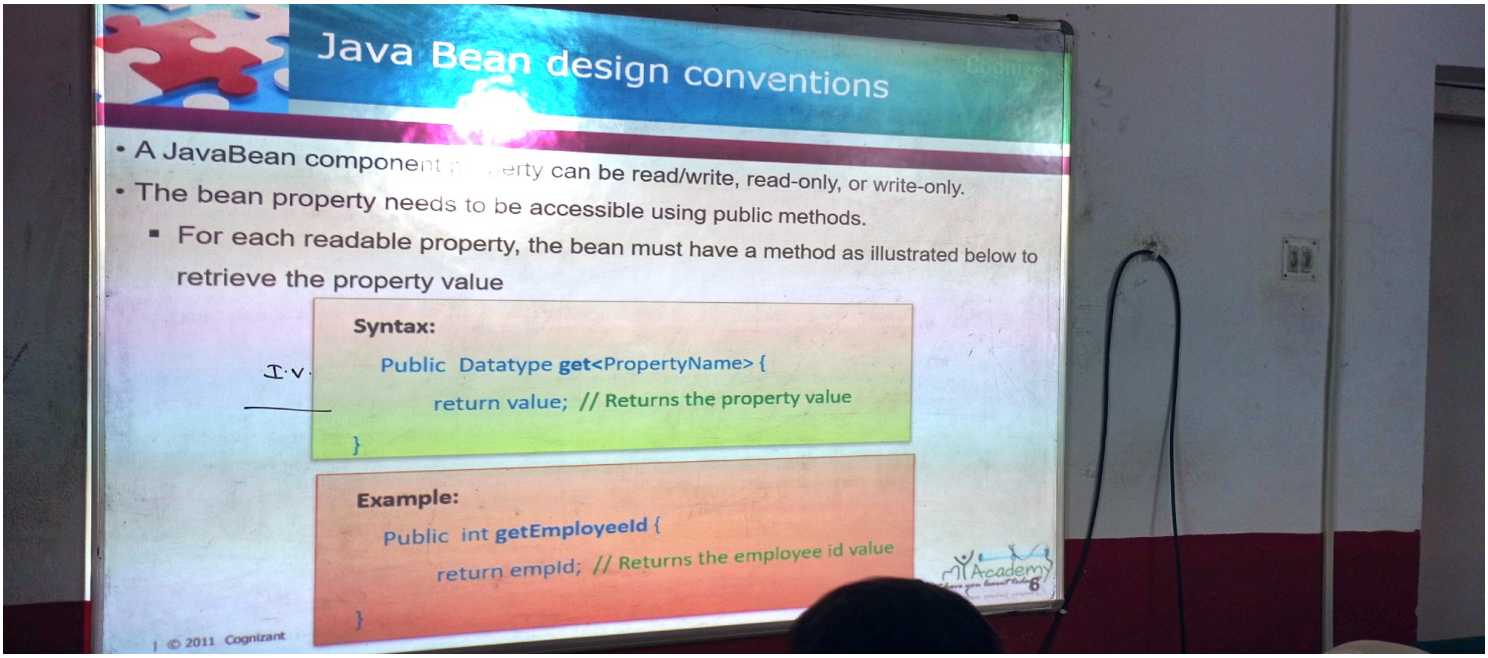
<div
style="height: 10%; bottom: 0; position: relative;"><%@include
file="footer.html"%>
</body>
</html>
isErrorPage attribute is set
Implicit object exception is available since isErrorPage attribute is set true.
© 2011 Cognizant

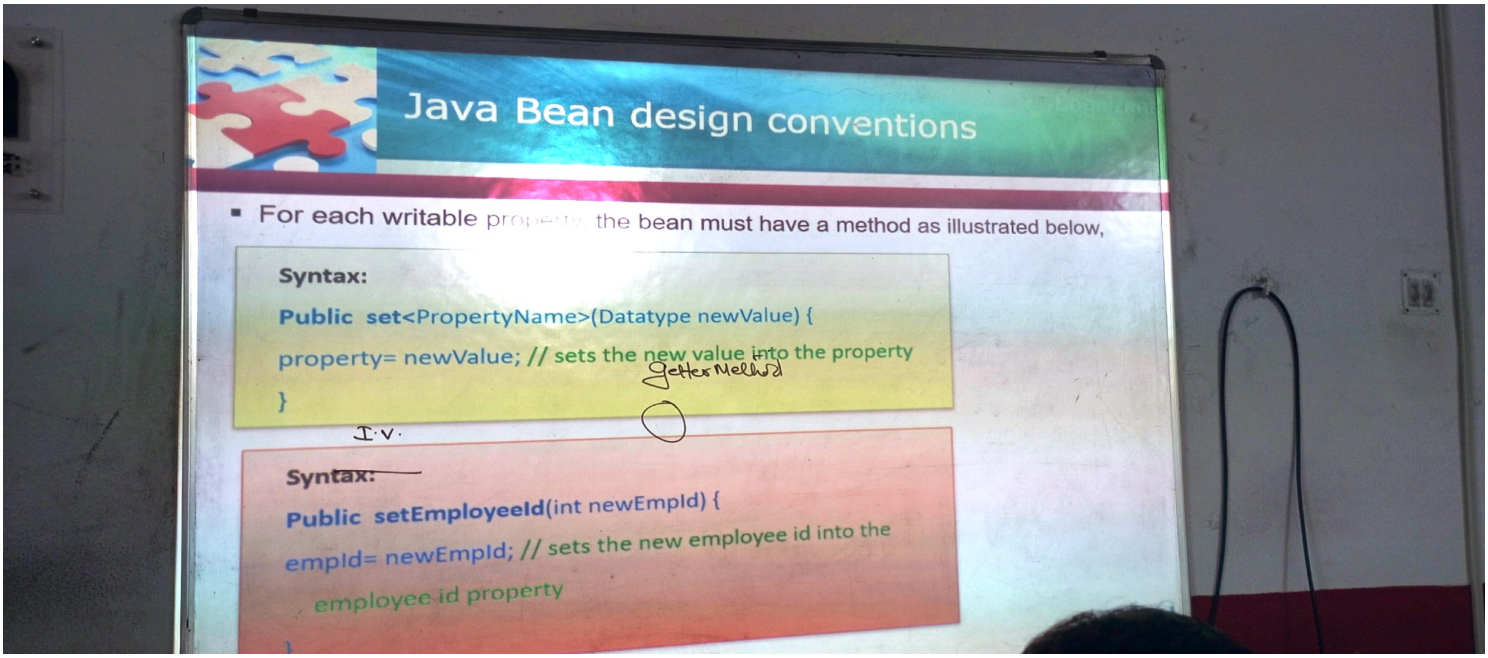

What is server side java bean ?
A server side java bean is a class used to store the details of real world entities
Example: Employee → Employee Name and Employee Salary,
Student → Student Name, Student Address.
Bean is a plain java class which contains
Fields (or) Properties: Fields to store the data, example: Employee Name, Salary.
Methods: Methods for retrieving and modifying the attributes like setEmployeeName(), setStudentAddress(). The methods are referred to as accessors/mutator.
Academy
5

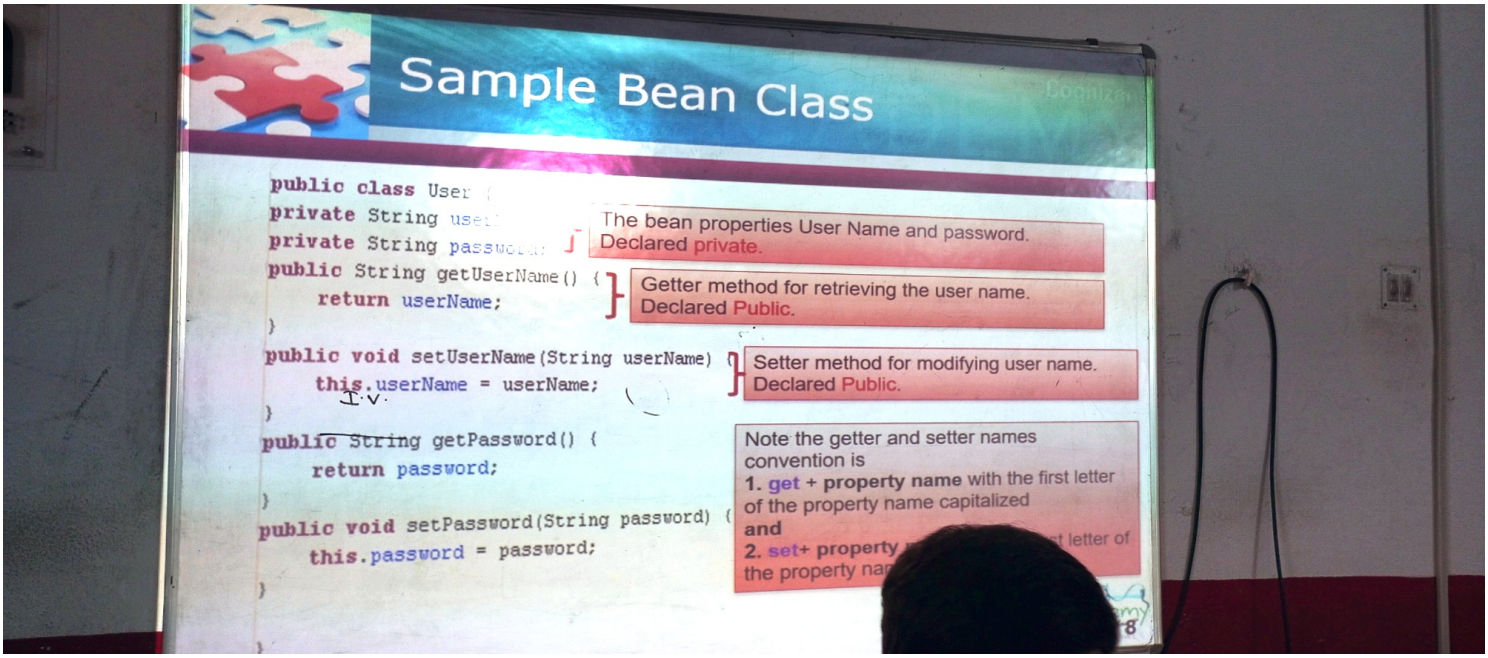

Java Bean design conventions
• A JavaBean component property can be read/write, read-only, or write-only.
• The bean property nee e accessible using public methods.
▪ For each readable property, the bean must have a method as illustrated below to retrieve the property value
Syntax:
Public Datatype get<PropertyName> {
return value; // Returns the property value
}

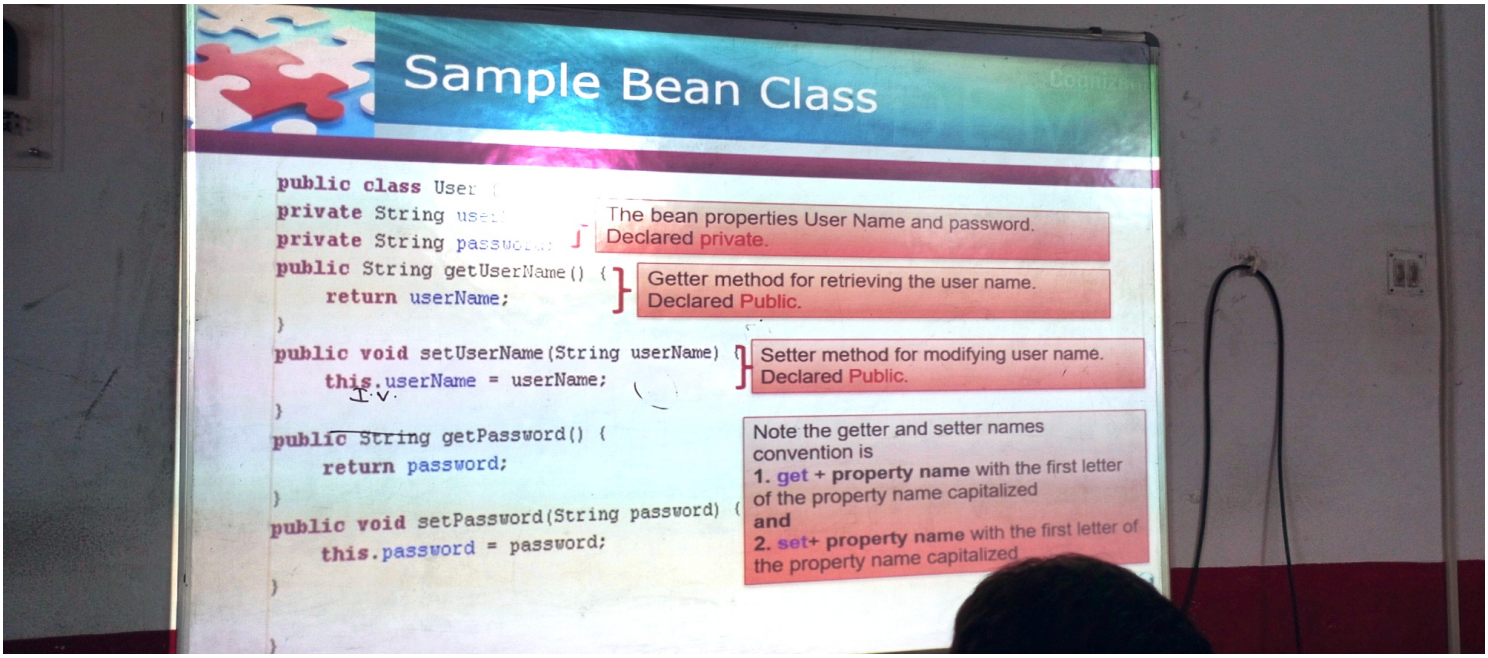

Java Bean design conventions
• A JavaBean component property can be read/write, read-only, or write-only.
• The bean property needs to be accessible using public methods.
▪ For each readable property, the bean must have a method as illustrated below to retrieve the property value
Syntax:
Public Datatype get<PropertyName> {
return value; // Returns the property value
}
Example:
Public int getEmployeeId {

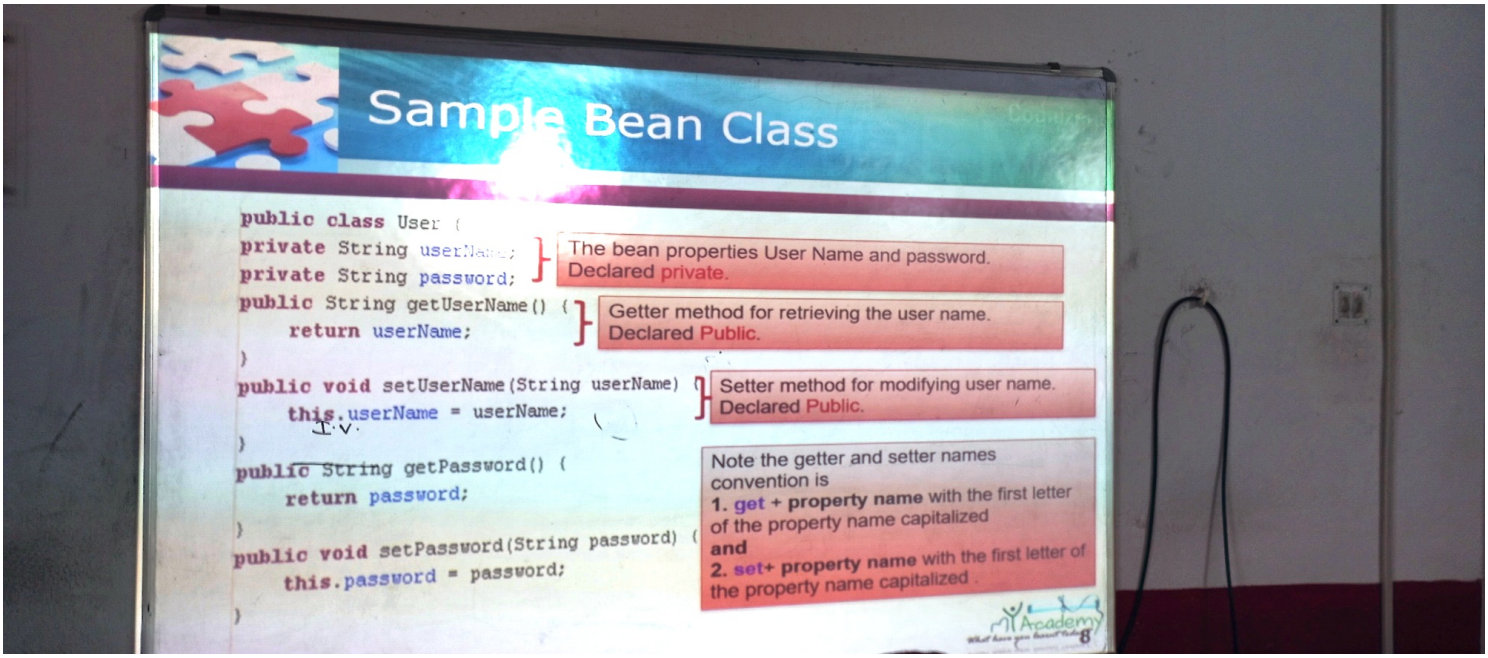

Java Bean design conventions
• A JavaBean component property can be read/write, read-only, or write-only.
• The bean property needs to be accessible using public methods.
▪ For each readable property, the bean must have a method as illustrated below to retrieve the property value
Syntax:
Public Datatype get<PropertyName> {
return value; // Returns the property value
}
Example:
Public int getEmployeeId {
return empId; // Returns the
id value

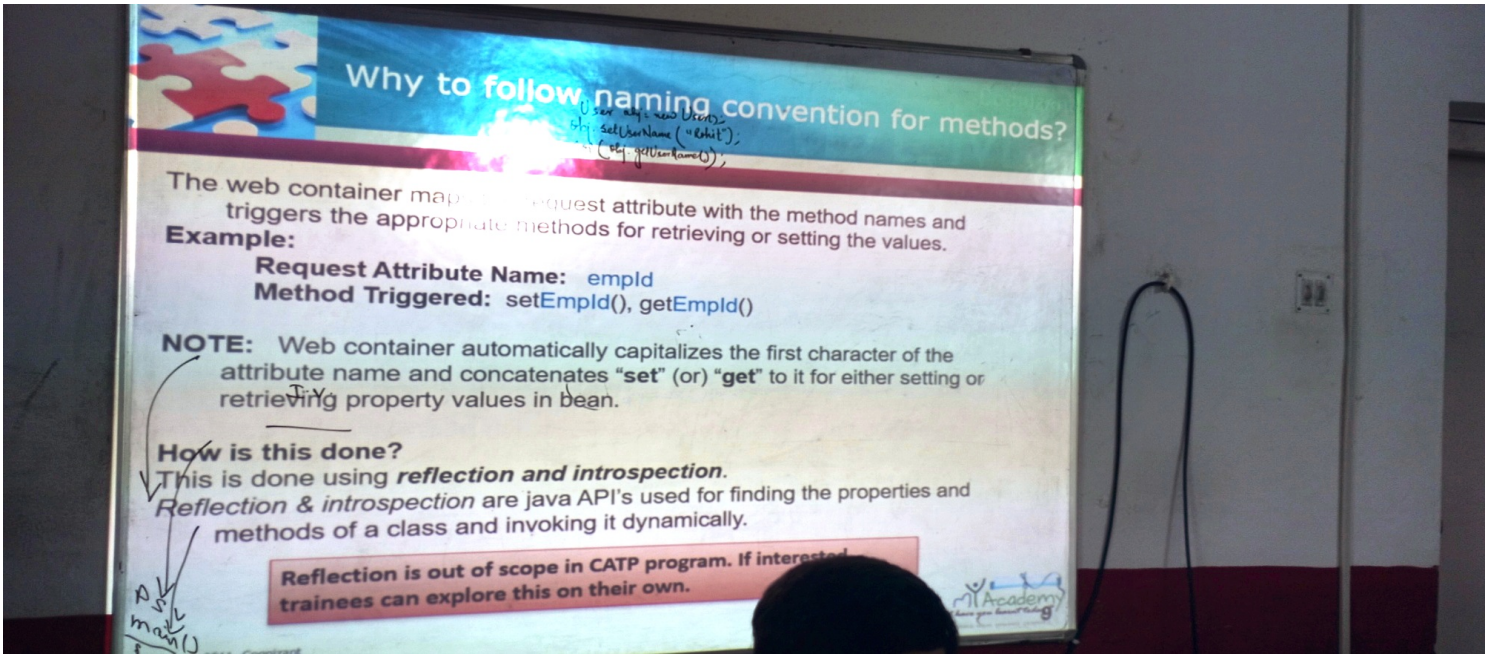

Java Bean design conventions
• A JavaBean component property can be read/write, read-only, or write-only.
• The bean property needs to be accessible using public methods.
▪ For each readable property, the bean must have a method as illustrated below to retrieve the property value
Syntax:
Public Datatype get<PropertyName> {
return value; // Returns the property value
}
Example:
Public int getEmployeeId {
return empId; // Returns the employee id value
}
© 2011 Cognizant
Academy
6

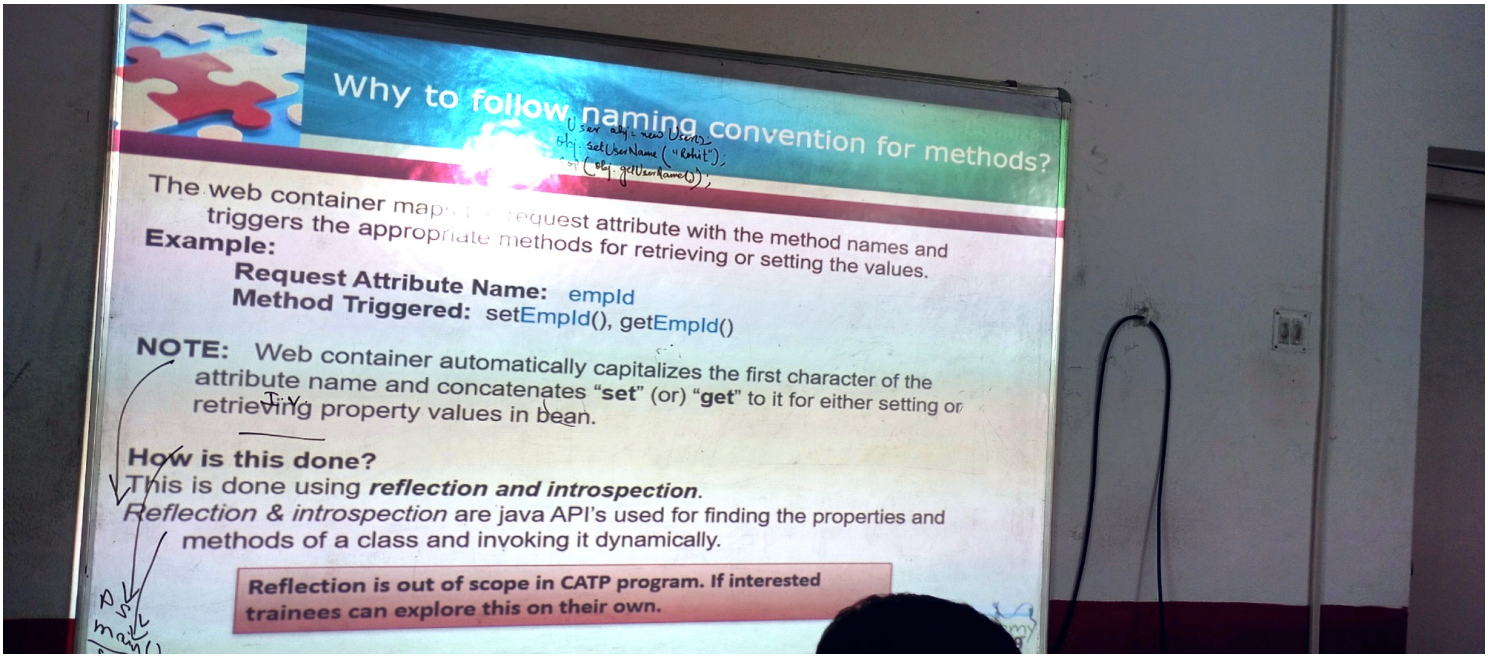

Java Bean design conventions
For each writable property the bean must have a method as illustrated below,
Syntax:
Public set<PropertyName>(Datatype newValue) {
property= newValue; // sets the new value into the property
}
Syntax:
Public setEmployeeId(int newEmpId) {
empId= newEmpId; // sets the new employee id into the
employee id property

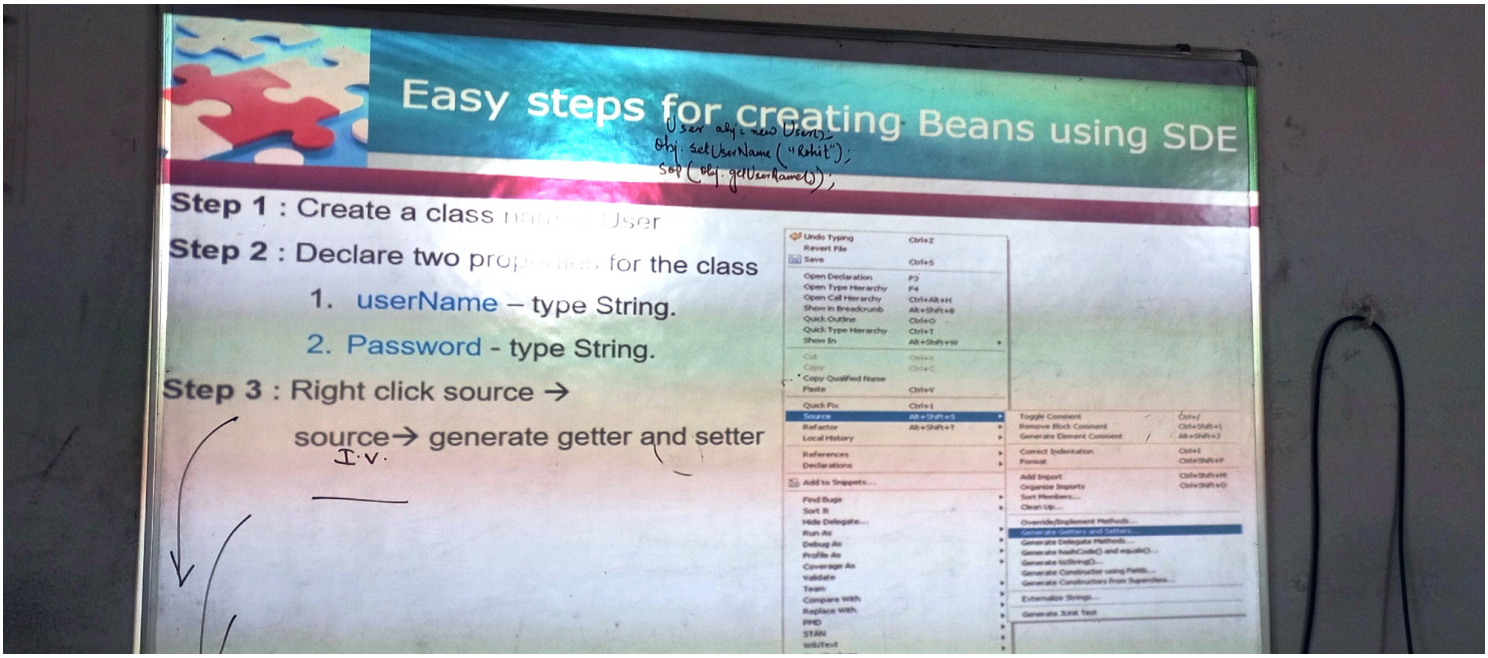

Sample Bean Class
public class User {
private String userName;
private String password;
public String getUserName() {
return userName;
}
public void setUserName(String userName) {
this.userName = userName;
}
public String getPassword() {
return password;
}
public void setPassword(String password) {
this.password = password;
}
}
The bean properties User Name and password.
Declared private.
Getter method for retrieving the user name.
Declared Public.
Setter method for modifying user name.
Declared Public.
Note the getter and setter names convention is
1. get + property name with the first letter of the property name capitalized
and
2. set+ property
the property na
letter of

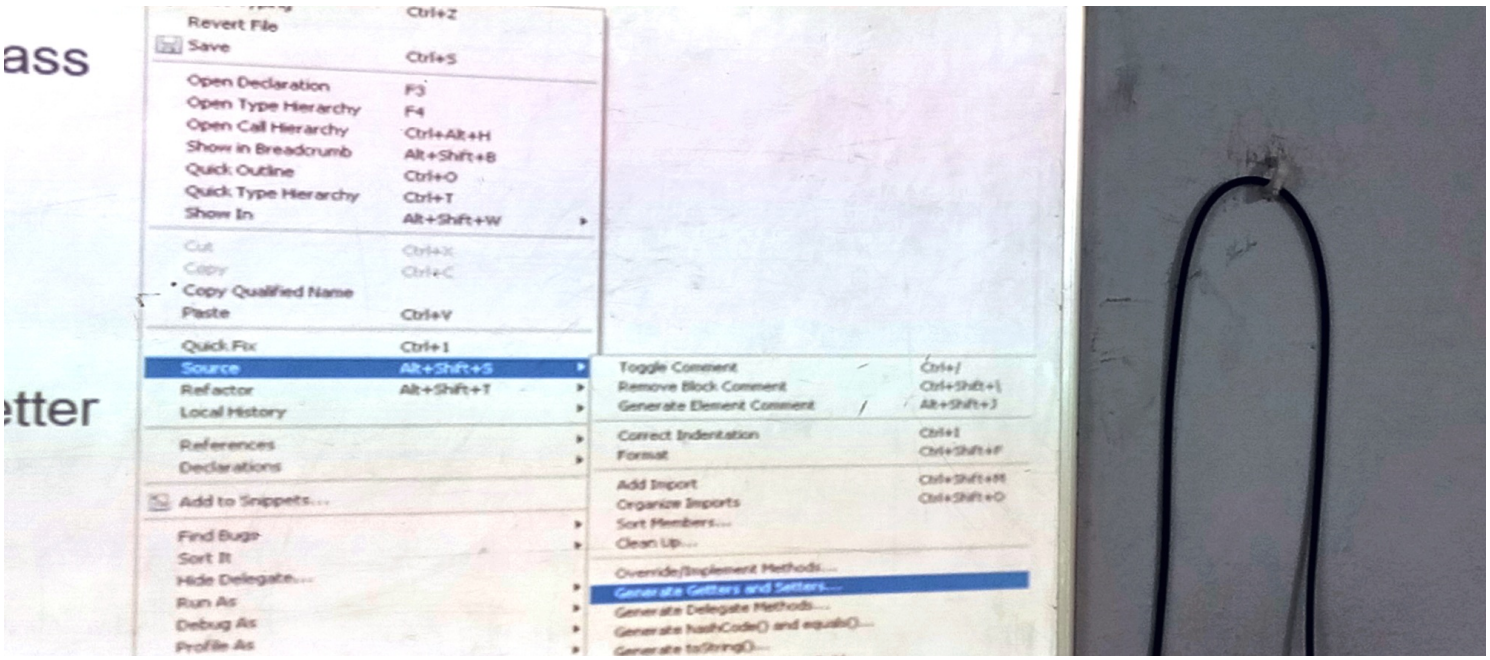

Sample Bean Class
public class User {
private String userName;
private String password;
public String getUserName() {
return userName;
}
public void setUserName(String userName) {
this.userName = userName;
}
public String getPassword() {
return password;
}
public void setPassword(String password) {
this.password = password;
}
}
The bean properties User Name and password.
Declared private.
Getter method for retrieving the user name.
Declared Public.
Setter method for modifying user name.
Declared Public.
Note the getter and setter names convention is
1. get + property name with the first letter of the property name capitalized
and
2. set+ property name with the first letter of the property name capitalized

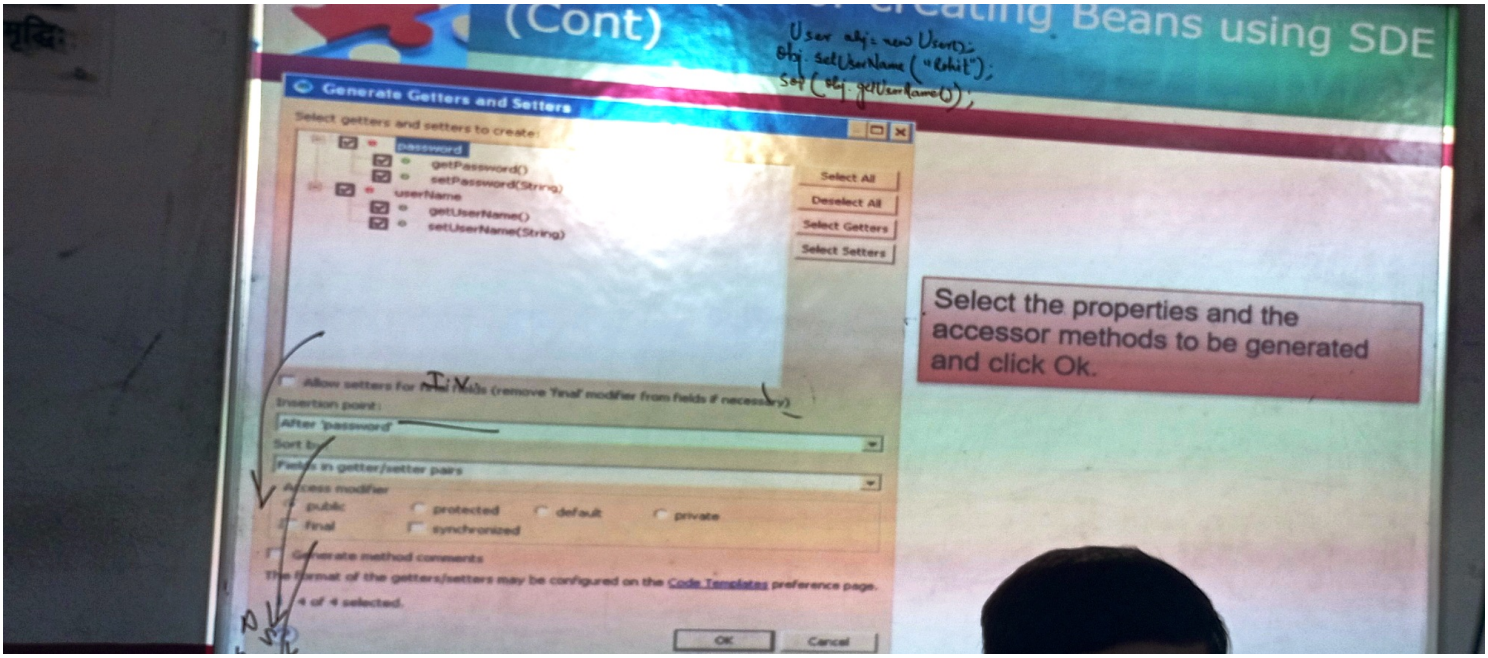

Sample Bean Class
public class User {
private String userName;
private String password;
public String getUserName() {
return userName;
}
public void setUserName(String userName) {
this.userName = userName;
}
public String getPassword() {
return password;
}
public void setPassword(String password) {
this.password = password;
}
The bean properties User Name and password. Declared private.
Getter method for retrieving the user name. Declared Public.
Setter method for modifying user name. Declared Public.
Note the getter and setter names convention is
1. get + property name with the first letter of the property name capitalized
and
2. set+ property name with the first letter of the property name capitalized .

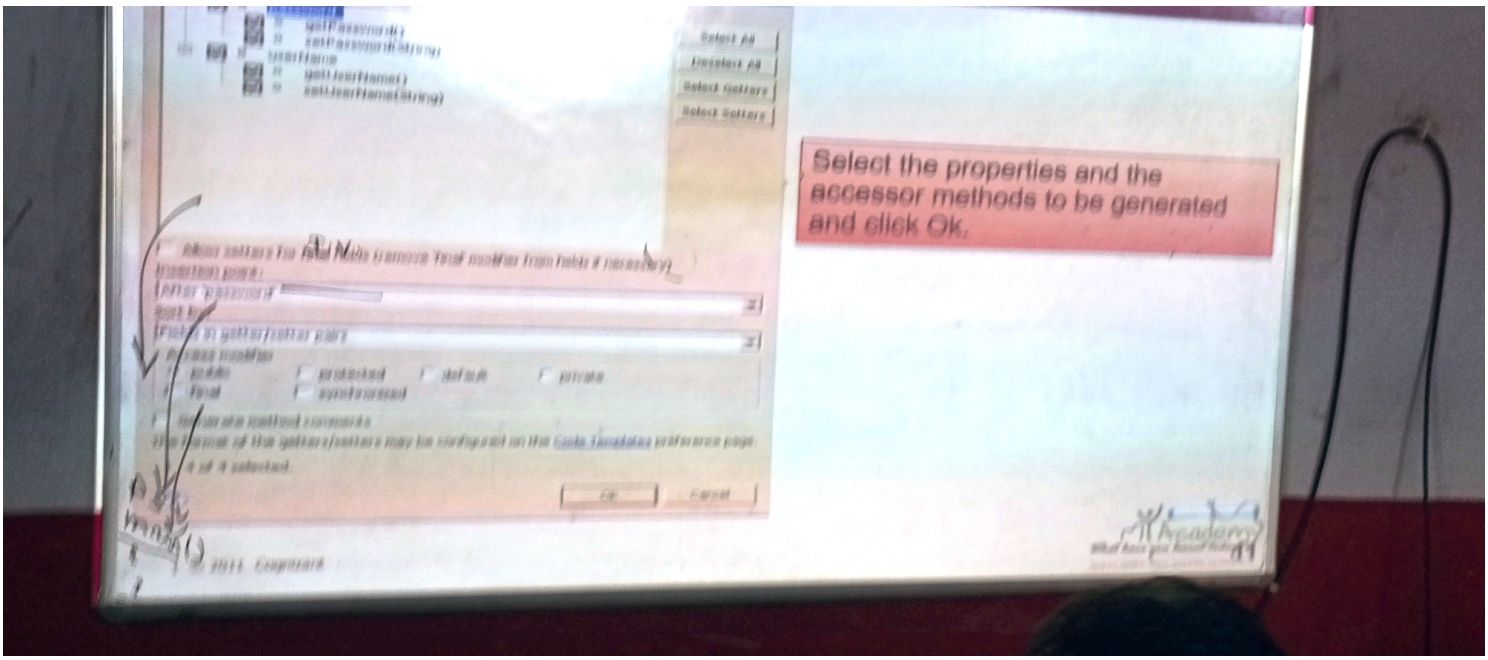

Why to follow naming convention for methods?
The web container maps the request attribute with the method names and triggers the appropriate methods for retrieving or setting the values.
Example:
Request Attribute Name: empId
Method Triggered: setEmpId(), getEmpId()
NOTE: Web container automatically capitalizes the first character of the attribute name and concatenates "set" (or) "get" to it for either setting or retrieving property values in bean.
How is this done?
This is done using reflection and introspection.
Reflection & introspection are java API's used for finding the properties and methods of a class and invoking it dynamically.
Reflection is out of scope in CATP program. If interested trainees can explore this on their own.

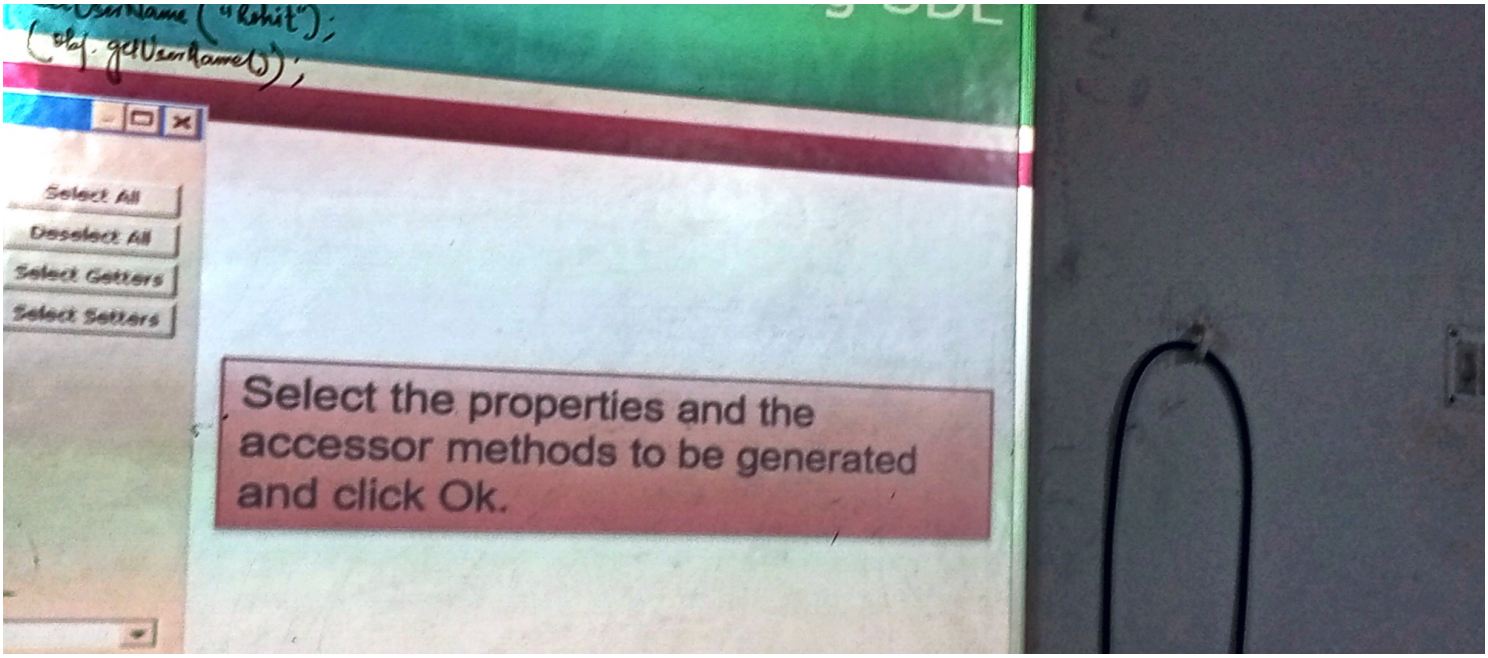

Why to follow naming convention for methods?
The web container maps the request attribute with the method names and triggers the appropriate methods for retrieving or setting the values.
Example:
Request Attribute Name: empId
Method Triggered: setEmpId(), getEmpId()
NOTE: Web container automatically capitalizes the first character of the attribute name and concatenates "set" (or) "get" to it for either setting or retrieving property values in bean.
How is this done?
This is done using reflection and introspection.
Reflection & introspection are java API's used for finding the properties and methods of a class and invoking it dynamically.
Reflection is out of scope in CATP program. If interested trainees can explore this on their own.

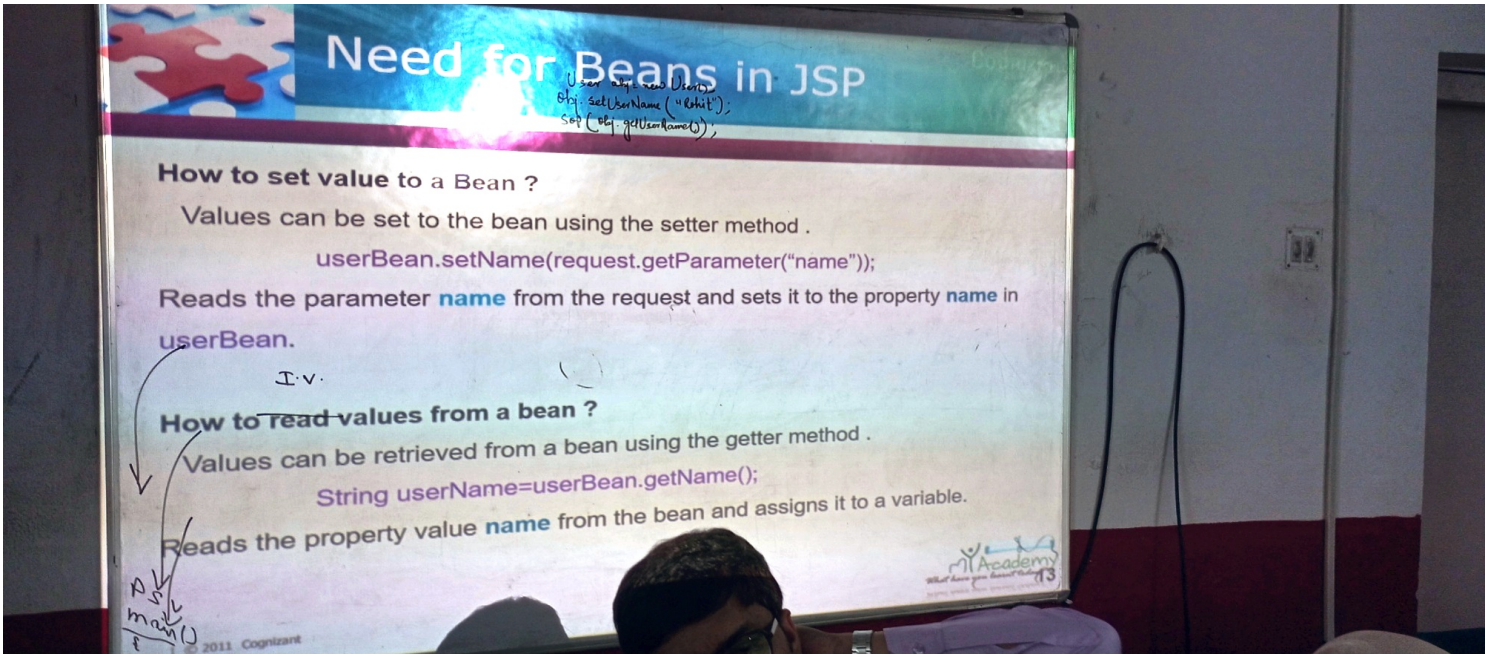

Easy steps for creating Beans using SDE
User obj = new User();
obj.setUserName("Rohit");
Sop(obj.getUserName());
Step 1 : Create a class
User
Step 2 : Declare two properties for the class
1. userName – type String.
2. Password - type String.
Step 3 : Right click source →
source→ generate getter and setter
I.V.

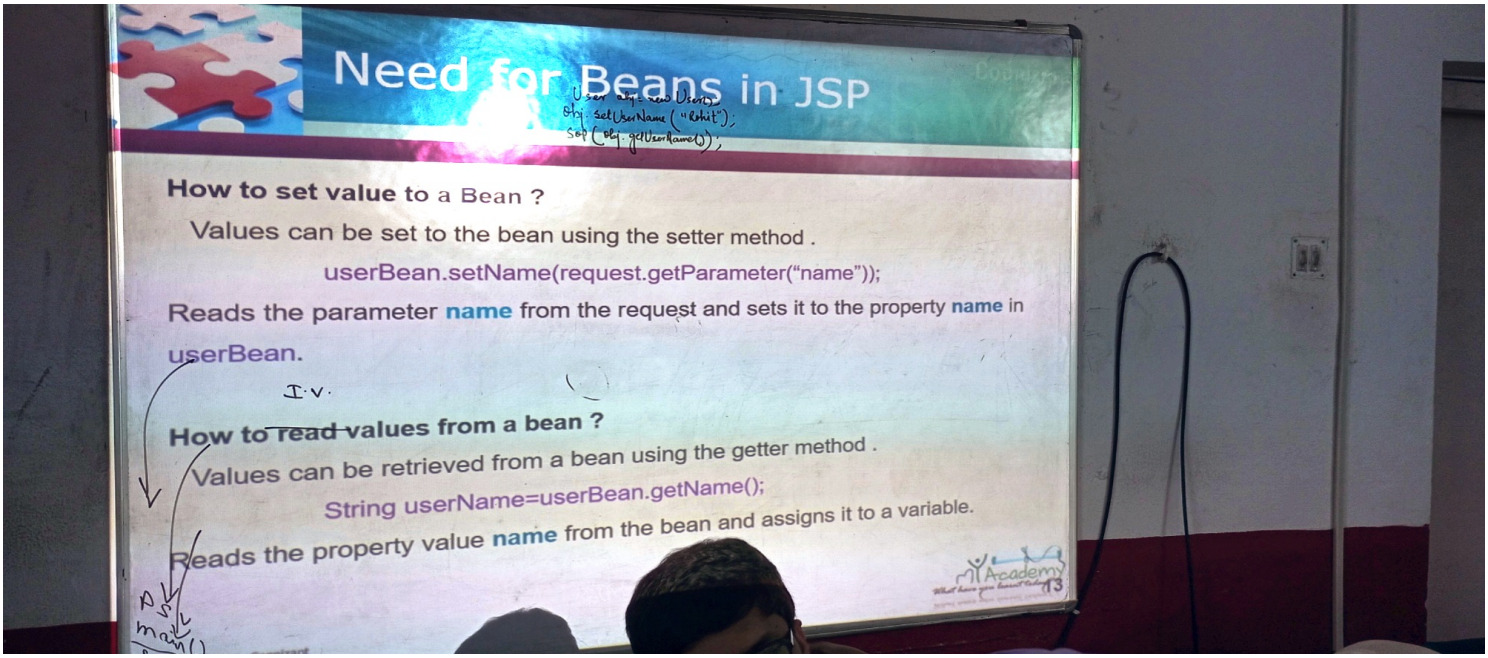

ass
etter
Revert File
Ctrl+Z
Save
Ctrl+S
Open Declaration F3
Open Type Hierarchy F4
Open Call Hierarchy Ctrl+Alt+H
Show in Breadcrumb Alt+Shift+B
Quick Outline Ctrl+O
Quick Type Hierarchy Ctrl+T
Show In Alt+Shift+W
Cut Ctrl+X
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Quick Fix Ctrl+1
Source Alt+Shift+S
Refactor Alt+Shift+T
Local History
References
Declarations
Add to Snippets...
Find Bugs
Sort It
Hide Delegate...
Run As
Debug As
Profile As
Toggle Comment Ctrl+/
Remove Block Comment Ctrl+Shift+\
Generate Element Comment Alt+Shift+J
Correct Indentation Ctrl+I
Format Ctrl+Shift+F
Add Import Ctrl+Shift+M
Organize Imports Ctrl+Shift+O
Sort Members...
Clean Up...
Override/Implement Methods...
Generate Getters and Setters...
Generate Delegate Methods...
Generate hashCode() and equals()...
Generate toString()...

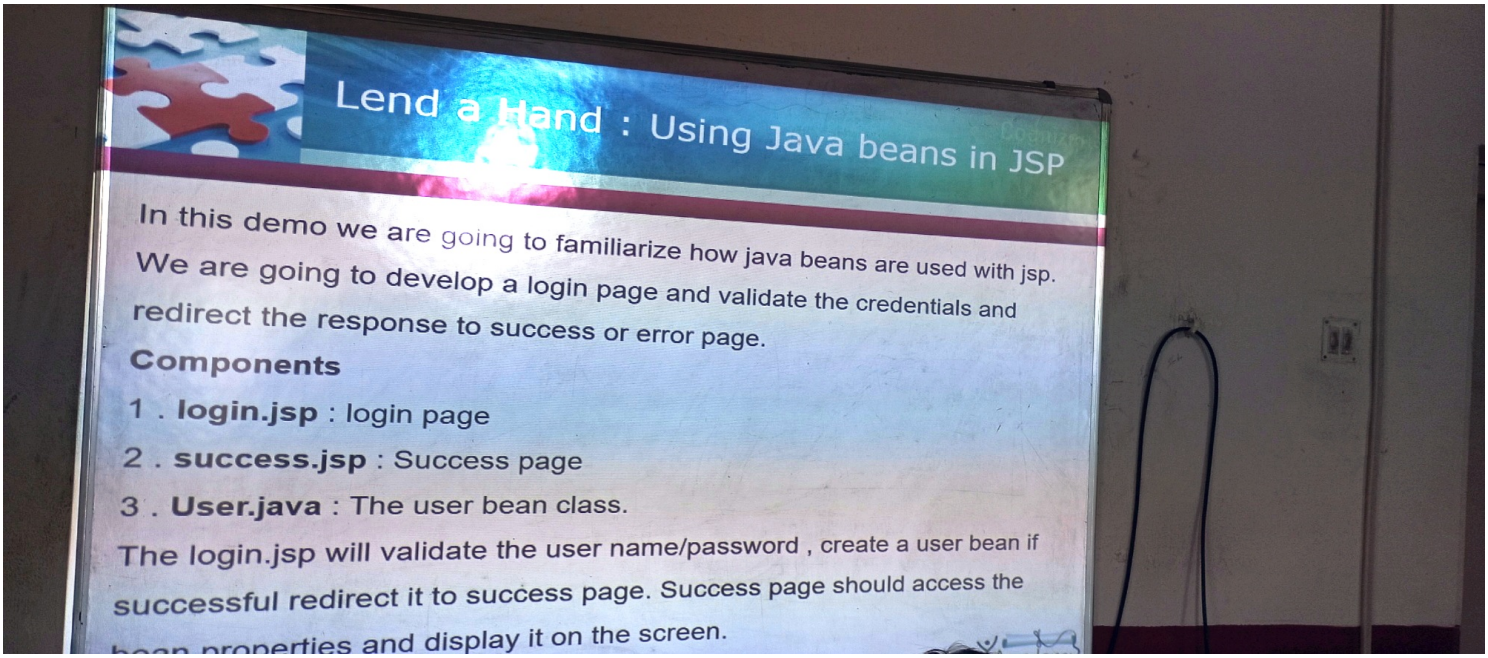

(Cont)
Beans using SDE
Generate Getters and Setters
Select getters and setters to create:
password
getPassword()
setPassword(String)
userName
getUserName()
setUserName(String)
Select All
Deselect All
Select Getters
Select Setters
Insertion point:
After 'password'
Access modifier
public
protected
default
private
final
synchronized
Generate method comments
The format of the getters/setters may be configured on the Code Templates preference page.
4 of 4 selected.
OK
Cancel
Select the properties and the accessor methods to be generated and click Ok.

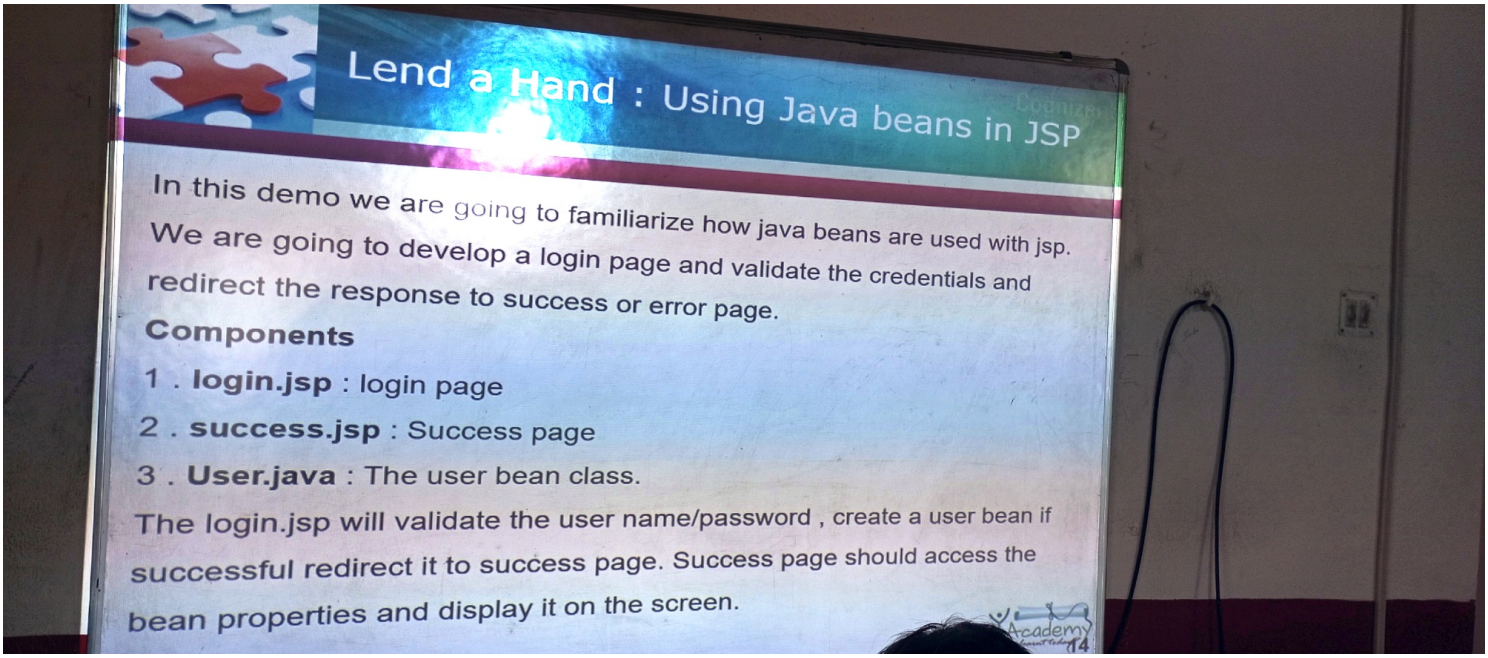

Select the properties and the accessor methods to be generated and click Ok.

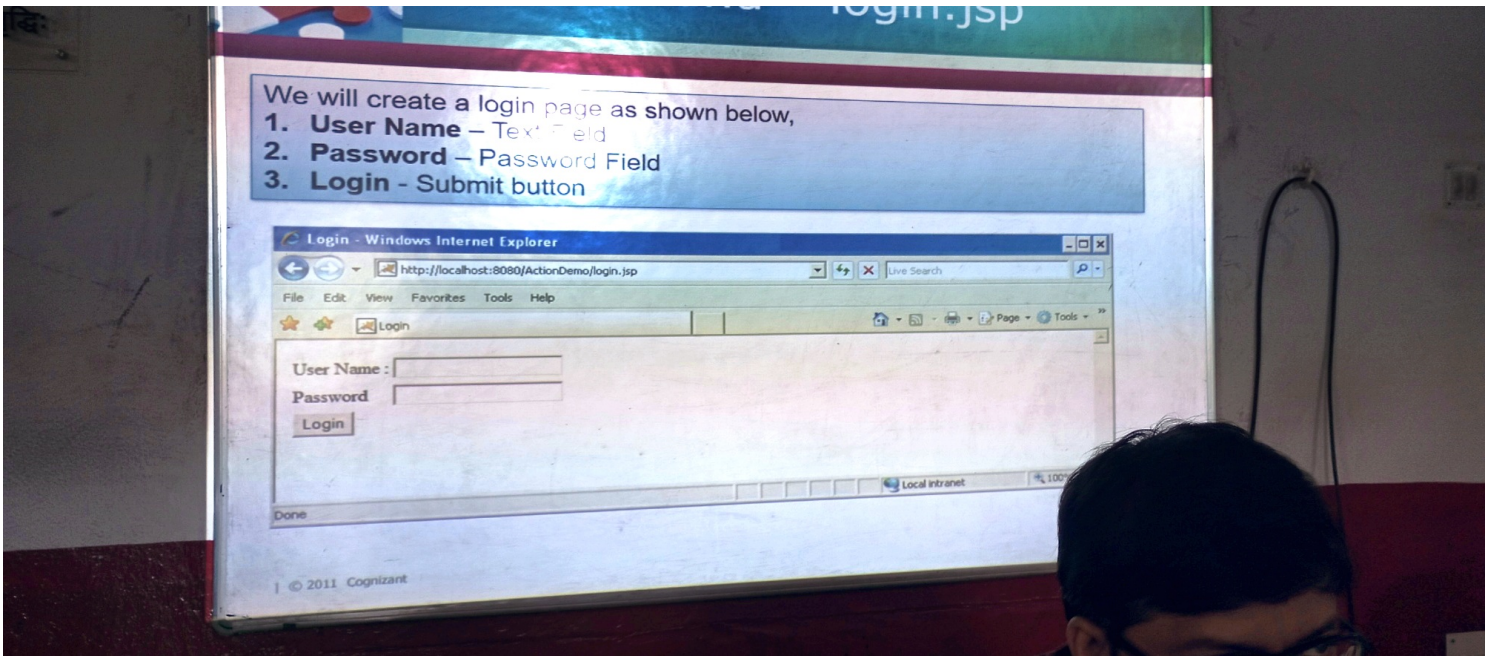

Select All
Deselect All
Select Getters
Select Setters
Select the properties and the accessor methods to be generated and click Ok.

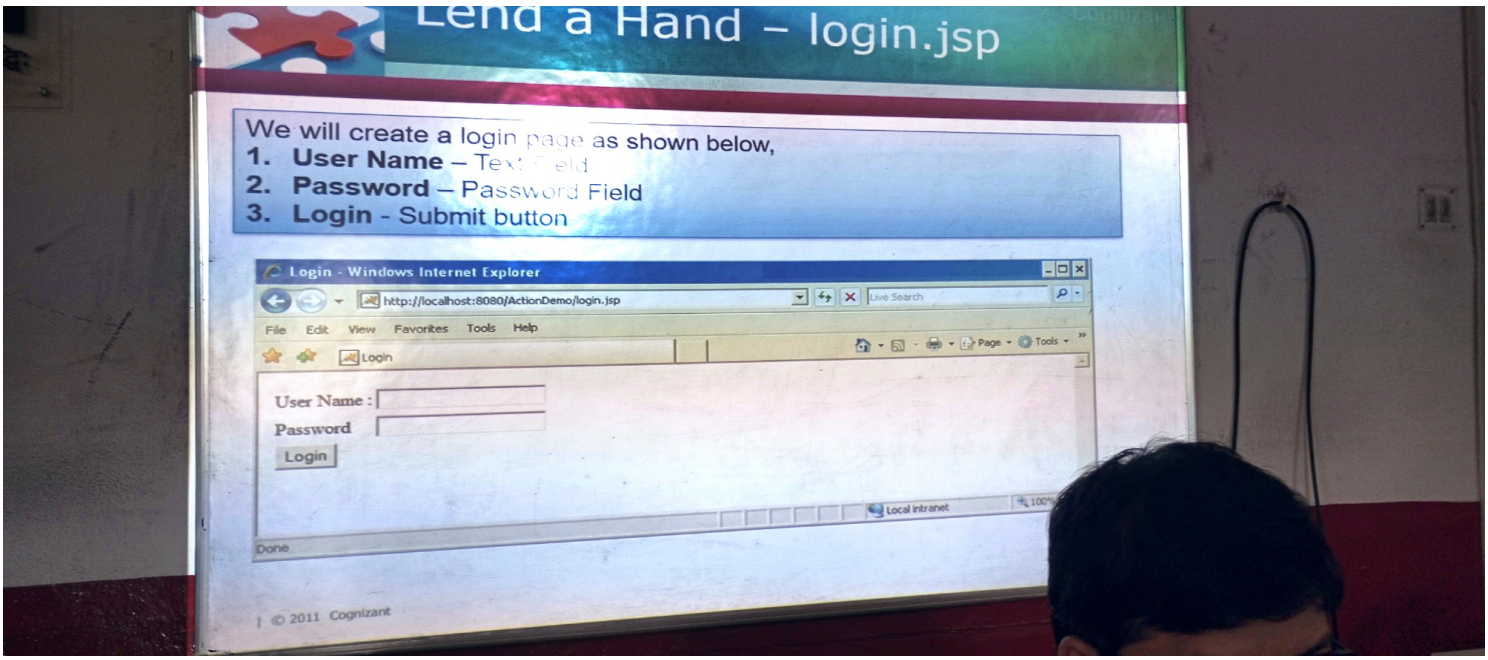

Need for Beans in JSP
How to set value to a Bean ?
Values can be set to the bean using the setter method .
userBean.setName(request.getParameter("name"));
Reads the parameter name from the request and sets it to the property name in userBean.
How to read values from a bean ?
Values can be retrieved from a bean using the getter method .
String userName=userBean.getName();
Reads the property value name from the bean and assigns it to a variable.
13
© 2011 Cognizant


Need for Beans in JSP
How to set value to a Bean ?
Values can be set to the bean using the setter method .
userBean.setName(request.getParameter("name"));
Reads the parameter name from the request and sets it to the property name in userBean.
How to read values from a bean ?
Values can be retrieved from a bean using the getter method .
String userName=userBean.getName();
Reads the property value name from the bean and assigns it to a variable.
13


Lend a Hand : Using Java beans in JSP
In this demo we are going to familiarize how java beans are used with jsp.
We are going to develop a login page and validate the credentials and
redirect the response to success or error page.
Components
1 . login.jsp : login page
2 . success.jsp : Success page
3 . User.java : The user bean class.
The login.jsp will validate the user name/password , create a user bean if
successful redirect it to success page. Success page should access the


Lend a Hand : Using Java beans in JSP
In this demo we are going to familiarize how java beans are used with jsp. We are going to develop a login page and validate the credentials and redirect the response to success or error page.
Components
1 . login.jsp : login page
2 . success.jsp : Success page
3 . User.java : The user bean class.
The login.jsp will validate the user name/password , create a user bean if successful redirect it to success page. Success page should access the bean properties and display it on the screen.


login.jsp
We will create a login page as shown below,
1. User Name – Text Field
2. Password – Password Field
3. Login - Submit button
Login - Windows Internet Explorer
http://localhost:8080/ActionDemo/login.jsp
File Edit View Favorites Tools Help
Login
User Name :
Password
Login
Done
© 2011 Cognizant


Lend a Hand – login.jsp
We will create a login page as shown below,
1. User Name – Text Field
2. Password – Password Field
3. Login - Submit button
Login - Windows Internet Explorer
http://localhost:8080/ActionDemo/login.jsp
File Edit View Favorites Tools Help
Login
User Name :
Password
Login
Local intranet
Done
© 2011 Cognizant