



Capstone Courseware, LLC

33 Boylston Street
Jamaica Plain, MA 02130

877-227-2477
capstonecourseware.com

Introduction to JSP with WebLogic

Robert J. Oberg
Will Provost

Instructor's Guide

Revision 10.3.0



Course Overview and Philosophy

Web development bridges two disciplines. The first is page design, in which HTML authors create professional quality pages for a web site. The second is programming, which involves implementing business logic, accessing databases, etc. – typically in Java. JSP facilitates a clean separation of these roles, and with the advent of native JSP expressions and JSTL this complete division of labor is a practical option.

This course aims to be largely accessible to both programmers who have a little web background and to page designers who know a little Java. That said, it's primary audience is page authors. Java coding is avoided completely, although there is some review of Java code provided with various exercises, just to clarify the role of Java classes in certain techniques.





Timeline

Day 1

Chapter 1	Web Applications
Chapter 2	JSP Architecture
Chapter 3	Scripting Elements
Chapter 4	Interactive JSP Applications
This may span Days 1 and 2	

Day 2

Chapter 5	Using JavaBeans
Chapter 6	The Expression Language and the JSTL
Chapter 7	Advanced JSP Features





Workshop Overlay

Capstone Courseware provides an optional package of workspace and project files for WebLogic Workshop for this course. (See the course Setup Guide for download URLs.) Instructors, use this package on your own initiative and at your own risk. You should have experience yourself with Workshop before using the overlay package in the classroom. The workspace and projects have been tested lightly with the course but are not part of the standard product.

And, the “win” in using Workshop for a bunch of JSP editing is probably mostly in the built-in JSP editor, which could be used without actually deploying the web application to a managed server within Workshop. The mode we recommend for using this material is to install and open the provided workspace, but then to use this only as a means to find, open, edit, and save JSP files, while managing a WebLogic server externally, from the command line, as described in the coursebook. This should give the best blend of editing facility and ease of testing, using an external browser.

But it is also possible, as with our other overlays, to perform nearly all the course tasks within the Workshop environment: start/stop a server, publish, test, debug, etc. The workspace has a built-in WebLogic server profile that assumes a domain exists at **c:\Capstone\JSP_WebLogic\WLDomain**. So, the best way to get started is to set environment and create the domain as described in the coursebook; then start Workshop and load the workspace; then start the server from within Workshop, and go from there.

Note that the server configurations are the same as in the Workshop workspaces for our servlets and JSTL courses. If you're teaching more than one of these modules, and don't want to have to swap workspaces, you can import the JSP and/or JSTL projects into the servlets workspace, deploy them to the **Servlets_WebLogic/WLDomain**, and test them from there.

Again, Capstone Courseware can only offer complete technical support on the standard course, and while we hope this overlay is convenient, it is not as thoroughly tested as the core lab image at this time. If a given exercise is giving trouble, please be certain to build and run it from the command line, using the SDK tools as prescribed in the student guide, before contacting Capstone.





Chapter 1. Web Applications

This chapter reviews the fundamentals of web technology, including HTML and HTTP. Various approaches to developing web applications are discussed, including CGI. The chapter provides a hands-on introduction to servlets and JSP with some small examples. The important points of this chapter are the Web technology review, introducing servlets and JSP in context, and establishing the authoring environment so that students can do hands-on work throughout the module.

Chapter 2. JSP Architecture

This chapter provides a conceptual overview of how servlets and JSP work. It also starts a systematic survey of the various kinds of JSP content and the most important implicit objects, such as **request** and **response**. JSP 2.0 has evolved into a broad diversity of possible syntax, and there are many ways to encode the same basic functions: scripting elements, standard actions, JSP expressions and JSTL actions overlap significantly in their capabilities. This chapter is partially meant to establish each of these and to distinguish them from one another ahead of time, so students don't start to get lost as they see multiple ways of solving a problem. Successive chapters will address each of these possible syntaxes.

Chapter 3. Scripting Elements

This chapter explains the uses of scripting elements: scriptlets, expressions and declarations. These are covered for the sake of completeness, but by the end of the module the use of scripts will have fallen from favor by contrast to the action- and expression-oriented authoring style encouraged by the JSP 2.0 specification. It might be a good idea to soft-peddle these techniques, so students (especially those with Java experience who might find scripts the most natural way to code pages) don't get too attached.

By the end of this chapter the student should already be able to do quite a lot of JSP development. The "odd and even" example gives a simple illustration of processing input data from clients by having the input supplied explicitly after a question mark on the URL. This provides a workable way to exercise some small JSPs and helps students without much Web background understand more about HTTP, without yet the details of HTML forms.





Chapter 4. Interactive JSP Applications

At this point the student already knows quite a lot about JSP, and the time has come to provide a more interesting example. The “electronic store” is a miniature e-commerce site. HTML forms are introduced for submitting data to the server, and JSP code provides handling of data. The application simulates purchasing of a single item, and provides motivation for session handling to implement a shopping cart. In this chapter the servlet session API is introduced, which is illustrated by a small program. The actual shopping cart example is postponed until the next chapter, where a simpler solution is provided. The chapter also includes a discussion of error handling, including an error page and explicitly throwing exceptions in your own Java code.

In the exercises that use the **errorPage** directive, the coursebook mentions a problem with using IE6. The longer story here is that IE6 shows what it calls “friendly” error messages by default: based only on the HTTP error code, it will show its own standard page, discarding whatever HTML is carried in the actual HTTP response body. This “helpful” feature gets in the way here; to turn it off, choose Tools|Options from the IE6 menu, select the Advanced tab, and under the Browsing category, see “Show friendly HTTP error messages” and uncheck this option. With this change IE6 will work correctly with JSP error pages, as other browsers will out of the box.

Chapter 5. Using JavaBeans

This is a key chapter, because it is with JavaBeans that some serious inroads can be made in removing business logic from the page itself. Actually, you can put business logic in ordinary Java classes, which can be called from simple scriptlet code or using standard actions.

Chapter 6. The Expression Language and the JSTL

The newest features of JSP are covered here: the native expression language and the use of JSTL. The students are well on their way now to seeing the whole 2.0 authoring style, in which business logic lives completely off-page, and even presentation logic can be packed into reusable fragments. Scripts are becoming a thing of the past.

While the EL is laid out in detail, only the most basic JSTL is covered; we’re just trying to understand the role of JSTL in page authoring and to learn to recognize its use.





Chapter 7. Advanced JSP Features

The course concludes with a chapter that briefly introduces several more advanced topics. The topic of custom tags is both very large and very important, and really warrants a course in its own right, especially as custom tag libraries are becoming more widely available. The chapter discusses both the more traditional approach of developing custom tags with Java handler classes and the newer, more facile technique of building tag files with snippets of callable JSP code.

The final topic that is discussed is threading. A simple example illustrates concretely how a race condition can arise on a JSP, and various approaches are discussed that will eliminate such a race condition. The example makes clear a fundamental difference between a JSP tag that separately declares a variable and scriptlet code that declares a local variable. You could mention this issue in the first module when both declarations and scriptlets are introduced, but it is probably better not to spend much time on the issue then. The Student Guide provides a simple forward reference to this last chapter.

Revision History

Revision 10.3.0 is the initial release of the course; it is based on Course 112, version 2.0.3.

Errata

At this time there are no errata for the course.





Feedback

We truly do welcome feedback, both of a specific nature (pointing out mistakes) and general suggestions. For the former sending email with a numbered list of corrections would be most helpful.

Please send feedback to:

Will Provost
Capstone Courseware
<mailto:provost@capstonecourseware.com>
www.capstonecourseware.com

