The Model-View-Controller (MVC) architecture, often
called the Model 2 architecture, has revolutionized web
application development. MVC allows a modular approach
to application building and a means by which JSP, servlets,
XML, XSLT, EJB, and more can work together. However, the cost of
constructing an MVC framework is still tremendously high.

Jakarta Struts makes that cost manageable by providing a robust, scalable,
open source MVC framework, complete with validation, error handling,
internationalization, and configurability. The Jakarta Struts Pocket
Reference gives you an inside track on getting Struts installed, configured,
and running smoothly in your enterprise. This book, an essential
companion to O’Reilly’s Programming Jakarta Struts, provides detailed
coverage of every Struts JSP tag, the Struts configuration files and
directives, built-in actions, and a wealth of other resources.

Chuck Cavaness is a Senior Technologist at the S1 Corporation. He’s the
author of Programming Jakarta Struts and has written articles for

Brian Keeton is a Java Architect at Visiprise, where he develops business
process and production management systems using J2EE. He is a Sun
Certified Java Developer and has taught courses in Java and CORBA.
## Contents

**Introduction**  1  

**Part I. Configuring Struts**  

- Configuring web.xml  4  
  - Configuring the Struts ActionServlet  5  
  - Declaring Initialization Parameters  6  
  - Using the load-on-startup Setting  7  
  - Setting Up the Welcome File List  8  
  - Configuring the Tag Libraries  9  

- The Struts Configuration Files  10  
  - Using Multiple Struts Configuration Files  10  
  - The data-sources Element  11  
  - The form-beans Element  13  
  - The global-exceptions Element  15  
  - The global-forwards Element  16  
  - The action-mappings Element  18  
  - The controller Element  21  
  - The message-resources Element  24  
  - The plug-in Element  25
Part II. Using Struts ActionForms

What Are ActionForms? 27
ActionForms and Scope 27
The Life Cycle of an ActionForm 28
Creating an ActionForm 28
  The validate( ) Method 31
  The ActionForm reset( ) Method 32
Declaring ActionForms in the Struts Configuration File 32
Using an ActionForm 33
The DynaActionForm Class 34
Configuring Dynamic ActionForms 35
DynaActionForm and Validation 37
  Creating the validation.xml File 37
  Plugging in the Validator 38

Part III. Built-in Action Classes

DispatchAction 39
ForwardAction 41
IncludeAction 42
LookupDispatchAction 43
SwitchAction 45
Part IV. Tag Reference

Conventions 47

Common Tag Attributes 48

Using JavaBeans with Struts Tags 49
  Accessing Simple Properties 49
  Accessing Nested Properties 49
  Accessing Indexed Properties 50

Bean Tags 50
  cookie 51
  define 51
  header 53
  include 53
  message 54
  page 55
  parameter 55
  resource 56
  size 57
 .struts 57
  write 58

HTML Tags 59
  Common HTML Attributes 61
  JavaScript Event Handlers 62
  HTML Tag Descriptions 63

Logic Tags 92
  Value Comparison 94
  Empty Value Check 95
  Value Present Check 96
  Substring Matching 97
Introduction

Jakarta Struts is an open source, Java™-based framework for building web applications. The framework was developed by the Apache Software Foundation and is housed online at http://jakarta.apache.org/struts/. Recognizing that every developer has a standard set of classes, tools, and utilities that are common to even the most divergent web applications, the Struts project has sought to take these commonalities and create from them a reusable and sturdy starting point for web application development.

From a technical standpoint, Struts is based upon the Model 2 approach, which is a slight variation on the Model-View-Controller (MVC) design pattern. Struts provides a generic controller that works well with any application, and then provides simple hooks into a variety of view technologies, such as JavaServer Pages, servlets, XSLT-based frameworks, and other Jakarta presentation frameworks such as Velocity templates. The controller can also easily interact with a variety of model components, including Enterprise JavaBeans™, JDBC, and object-relational mapping technologies. It is no exaggeration to say that Jakarta Struts can handle nearly any combination of model and view technologies you could dream up—and it handles the task seamlessly.
While there are literally hundreds of ways to build a web application, Jakarta Struts takes the most common components of all these solutions and provides them to the developer, prebuilt, documented, and ready for use. Logging, internationalization, debugging, validation, and more are all provided with a simple download of the Struts framework. It is no wonder that Struts (and the books that document it) are becoming the hottest thing going in the Java technology space.

However, as with any complex framework, there is more to using it than binary objects, source code, or even comprehensive user manuals. There are those repetitive tasks that involve this particular method, or that particular set of arguments, that take up space in the memory banks of every developer. As you begin to take advantage of Struts, from its simplest to its most complex functionality, you will find yourself trying to remember exactly which tags do what and where to place that XML element in your Struts configuration file. While you could certainly page through a 500 page tome on Jakarta Struts and find your answer, this book takes a more practical approach (and one that is quite a bit smaller!). The authors have provided a quick reference to all the common tasks you’ll find in Struts, allowing you to take nothing more than your Struts experience and this book in a back pocket to client engagements. Whether it’s a default action class or the location of Struts add-on classes, this pocket reference has it all, in a size that even today’s airlines won’t give you trouble over. We hope you’ll enjoy the reference and find it helpful in your daily Struts programming.

**Conventions Used in This Book**

The following typographical conventions are used in this book:
**Italic**  
Used to indicate new terms, URLs, filenames, file extensions, and directories.

**Constant width**  
Indicates code lines that should be typed verbatim; names and keywords in Java programs, including method names, variable names, and class names; and XML element names, tags, attributes, and values.

**Constant width bold**  
Used in code examples to emphasize lines of particular note.

**Constant width italic**  
Used in examples and tables to show text that should be replaced with user-supplied values.

---

**NOTE**  
Indicates a tip, suggestion, or general note.

---

**WARNING**  
Indicates a warning or caution.
PART I

Configuring Struts

For every Struts application, there are at least two configuration files that must be present: the web application deployment descriptor and a Struts configuration file. This part discusses each one in turn.

Configuring web.xml

Each Struts application must include a web application deployment descriptor named `web.xml`, which must be placed in the `WEB-INF` directory. The web container reads and parses the descriptor file at startup and uses the settings to configure the runtime environment for the installed web application.

Although there are many available configuration settings that can affect the container’s runtime environment, it’s not necessary to configure all of the settings for a Struts application. In many cases, the absence of a setting or the container’s default values will be sufficient. Only those settings that pertain to Struts applications will be examined here.

For Struts applications, the following configuration settings are typically configured within the `web.xml` file:

- Struts ActionServlet settings
- Initialization parameters
- load-on-startup settings
- Welcome file list
- Tag libraries mappings
Configuring the Struts ActionServlet

The Struts ActionServlet is designed to receive all incoming requests for the web application. Two steps are necessary when configuring the ActionServlet in the web.xml file. The first step is to use the servlet element to configure the fully qualified Java class name of the ActionServlet:

```xml
<web-app>
  <servlet>
    <servlet-name>storefront</servlet-name>
    <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
  </servlet>
</web-app>
```

In this web.xml example, the servlet element declares two child elements, servlet-name and servlet-class. The servlet-class element specifies the fully qualified class that will function as the front controller for the Struts application. The Java class specified must be a descendant of the org.apache.struts.action.ActionServlet. If you don’t have a need for specialized behavior, you can safely use the default controller as shown in the previous web.xml fragment.

The servlet-name element acts as a logical name for the ActionServlet. It is used in other elements within the deployment descriptor. You can specify whatever value you like here, as long as it adheres to the Servlet Specification naming guidelines.

The second step required to configure the ActionServlet is to inform the web container which URL requests should be directed to the ActionServlet controller. This is done using the servlet-mapping element:

```xml
<web-app>
  <servlet>
    <servlet-name>storefront</servlet-name>
    <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>storefront</servlet-name>
    <url-pattern>/storefront/*</url-pattern>
  </servlet-mapping>
</web-app>
```
<servlet-mapping>
  <servlet-name>storefront</servlet-name>
  <url-pattern>*.do</url-pattern>
</servlet-mapping>
</web-app>

In the example servlet and servlet-mapping elements shown, any request containing a URL that matches ".do" would be processed by the servlet named storefront.

**Declaring Initialization Parameters**

Within the servlet element, you can specify multiple init-param elements. The parameters listed in Table 1 are used by the ActionServlet to configure the Struts application runtime environment.

<table>
<thead>
<tr>
<th>Name</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>config</td>
<td>Comma-separated list of context-relative path(s) to the XML resource(s) containing the configuration information for the default module. The default value is /WEB-INF/struts-config.xml.</td>
</tr>
<tr>
<td>config/${module}</td>
<td>Comma-separated list of context-relative path(s) to the XML resource(s) containing the configuration information for the module that will use the specified prefix (/${module}). This can be repeated as many times as required for multiple modules.</td>
</tr>
<tr>
<td>convertNull</td>
<td>Forces simulation of the Struts 1.0 behavior when populating forms. If set to true, the numeric Java wrapper class types such as java.lang.Integer default to null (rather than 0). The default value is false.</td>
</tr>
<tr>
<td>rulesets</td>
<td>Comma-delimited list of fully qualified class names of additional org.apache.commons.digester.RuleSet instances that should be added to the Digester and that will process the Struts configuration file. By default, the RuleSet for the standard configuration elements is loaded.</td>
</tr>
<tr>
<td>validating</td>
<td>Specifies whether a validating XML parser should be used to process the configuration file. The default value is true.</td>
</tr>
</tbody>
</table>
R
radio tag
   HTML library, 60, 86–88
   Nested library, 110
redirect tag (Logic library), 93
   attributes of, 100
request parameters, 31, 43–45
request scope, 27
   reset() and, 32
RequestDispatcher class, 43
reset tag (HTML library), 60
   attributes of, 88
reset(), 32
   checkbox tag and, 65
   configuring
      DynaActionForms, 36
      multibox tag and, 80
   select tag and, 89
resource bundles, 24, 43
   format/formatKey attributes
      for write tag, 59
resource tag (Bean library), 51
   attributes of, 56
resources
   discovering dynamically at
      startup, 25
   forwarding to, 41
   including output of, 42
   information on Struts, 125
   mapping logical names to, 17
rewrite tag (HTML library), 60
   attributes of, 88
root tag (Nested library), 106
   rulesets initialization
      parameter, 6
S
scope and ActionForms
   scope attribute
      common tag attribute, 48
      specifying for action
         mappings, 33
select tag
   HTML library, 60, 89
   Nested library, 110
   servlet element, 5–7
   servlet-class element, 5
   load-on-startup element, 7
   ServletException error, 15
   servlet-mapping element, 5
   servlet-name element, 5
session scope, 27
   reset() and, 32
set-property element
   controller element, 21
   data-source element, 11–13
   message-resources
      element, 24
   plug-in element, 26
   simple properties, accessing, 49
size tag
   Bean library, 51, 57
   Nested library, 110
Struts
   ActionServlet
      configuring, 5
      initialization parameters
         for, 6
      configuring, 4–26
   information resources for, 125
   mailing lists, 125
   Struts Console tool, 126
   Struts Resource Page, 126
   struts tag (Bean library), 51
   attributes of, 57
   struts-config.xml file, 10, 45
   STRUTS-DEV mailing list, 125
   STRUTS-USER mailing list, 125
   style attribute (common
      HTML tag attribute), 61
   styleClass attribute (common
      HTML tag attribute), 61
   styleId attribute (common
      HTML tag attribute), 61
   submit buttons on forms, 43–45
submit tag
HTML library, 60, 90
Nested library, 110
substring matching tags, 97
SwitchAction class, 39, 45

T

tabindex attribute (common HTML tag attribute), 61
tabs layout, 112
tag libraries
Bean, 50–59
custom attributes for HTML tags, 61
custom attributes for tags, 48
configuring, 9
HTML, 59–92
JavaBeans and, 49
Logic, 92–104
Nested, 104–111
Tiles, 111–124
taglib element, 9
text tag
HTML library, 61, 91
Nested library, 110
textarea tag
HTML library, 61, 91
Nested library, 110
Tiles tag library, 111–124
title attribute (common HTML tag attribute), 61
titleKey attribute (common HTML tag attribute), 61
tools for Struts, 126
toString(), 58
if collection attribute not specified, 88
getAsNesting tag, 88
link tag and, 77
substring matching, 98
type, casting ActionForm to proper, 34

U

update(), 41
useAttribute tag (Tiles library), 115
attributes of, 124

V

validate attribute, specifying for action mappings, 33
validate(), 20, 31, 65
validating
ActionForm class, 29–32
DynaActionForm class, 37
validating initialization parameter, 6
validation methods, rendering, 75
validation.xml file, creating, 37
Validator framework, 35, 37
value comparison tags, 94
value present check tags, 96
vertical box layout, 112

W

web.xml file, configuring, 4–10
WEB-INF directory, 4, 10
web site for Struts information, 126
welcome-file-list element, 8
write tag
Bean library, 51, 58
Nested library, 110
writeNesting tag (Nested library), 110

X

xhtml tag (HTML library), 92