International Journal of Research (IJR) Vol-1, Issue-9, October 2014 **ISSN 2348-6848**

STRUTS

Bibhuti Bhusan; Harsh Chawla; Ashutosh Bhatt

Abstract:

Java is a computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that code that runs on one platform does not need to be recompiled to run on another. Java applications are typically compiled to bytecode (class file) that can run on any Java virtual machine (JVM) regardless of computer architecture. Java is, as of 2014, one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by James Gosling at Sun Microsystems (which has since merged into Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

Keywords:-JVM; WORA; JSP; HTML; MVC

Introduction:

In this I am going to illustrate the functioning of Struts in detail. When Java servlets was invented, many coders realized that it was a Good Thing. It was faster and powerful then the standard CGI and was portable, and infinitely extensible.

But coding in HTML to send to browser in endless println() statements was problematic. The answer was JavaServer Pages, which turned Servlet writing inside-out. Now developers mix HTML with Java code, and have advantages of servlets.

Java web applications quickly became "JSP-centric". This in-and-of itself was not a Bad Thing, but it did little to resolve flow control issues and other problems endemic to web applications.

Clearly, another paradigm needed soon.

Many coder realized that JSP and servlets could be used together to deploy web applications. The servlets could help with the control-flow, and the JSPs could focus on the nasty business of writing HTML. In due course, using JSPs and servlets together became known as Model 2 (meaning, presumably, that using JSPs alone was Model 1).

There was nothing new under the Sun ... and many of them have been quick to point out that JSP's Model 2 follows the classic MVC design pattern abstracted from the venerable Smalltalk MVC framework. Java Web developers now tend to use the terms Model 2 and MVC interchangeably.

"What Is Struts and Why Should I Care?"

Struts is an application development framework that is designed for and used with the popular J2EE (Java 2, Enterprise Edition) platform. It cuts time out of the development process and makes developers more productive by providing them a series of tools and components to build applications with. It is non-proprietary and works with virtually any J2EE-compliant application server. Struts falls under the Jakarta subproject of the Apache Software Foundation and comes with an Open Source license (meaning it has no cost and its users have free access to all its internal source code).

HOW important struts is:

Using a framework mean that you don't have to spend time building your entire application, you can focus on coding the business logic and the presentation layer of the application—not the overhead pieces like figuring out how to capture user input or figuring out how to generate drop-down boxes on a Web page.

Using a framework also helps you encode best practices. The framework developers have put a lot of thought into the best approaches to application building—why reinvent this yourself?

Another benefit of using a framework is that it allows your code (at least in the case of Struts) to be highly platform independent. For example, the same Struts code should work under Tomcat on an old Windows machine as runs using Weblogic on Linux or Solaris in production. And this can be accomplished without even recompiling in many cases—the same Web application (or ". war" file) can simply be copied from one server to another.

Another extremely important benefit—especially if you're relatively new to Web development—is that it gives you a place to start. Any developer will tell you it's easier to take a basic application and modify it than it is to build something from scratch. This feature of Struts can save you days or weeks of planning and development.

Today, I create virtually nothing from scratch. Almost no one who is an experienced developer does. In fact, some of the greatest successes in software development were based on this exact idea. For example, in 1991 when Linus Torvalds began building the operating system that today is Linux, he began with the operating system Minix. He got a copy of the Minix source code, looked it over in detail, and used it as the basis for Linux. And while the first launch of Linux contained none of the original Minix code, Linus surely went further, faster because he had it to start with.

How Does Struts Work?

Struts is based on the time-proven Model-View-Controller (MVC) design pattern. The MVC pattern is widely recognized as being among the most well-developed and mature design patterns in use. By using the MVC design pattern, processing is broken into three distinct sections aptly named the Model, the View, and the Controller. These are described in the following subsections:

Model Components

Model components provide a "model" of the business logic or data behind a Struts program. For example, in a Struts application that manages customer information, it may be appropriate to have a "Customer" Model component that provides program access to information about customers.

It's very common for Model components to provide interfaces to databases or back-end systems. For example, if a Struts application needs to access employee information that is kept in an enterprise HR information system, it might be appropriate to design an "Employee" Model component that acts as an interface between the Struts application and the HR information system.

Model components are generally standard Java classes. There is no specifically required format for a Model component, so it may be possible to reuse Java code written for other projects.

View Components

View components are those pieces of an application that present information to users and accept input. In Struts applications, these correspond to Web pages.

View components are used to display the information provided by Model components. For example, the "Customer" Model component discussed above would need a View component to display its information. Usually, there will one or more View components for each Web page in a Struts application.

View components are generally built using JavaServer Page (JSP) files. Struts provides a large number of "JSP Custom Tags" (sometimes referred to as Struts Tags) which extend the normal capabilities of JSP and simplify the development of View components.

Controller Components

Controller components coordinate activities in the application. This may mean taking data from the user and updating a database through a Model component, or it may mean detecting an error condition with a back-end system and directing the user through special error processing. Controller components accept data from the users, decide which Model components need to be updated, and then decide which View component needs to be called to display the results.

One of the major contributions of Controller components is that they allow the developer to remove much of the error handling logic from the JSP pages in their application. (After all, if errors in processing occur, the Controller component forwards to an error-processing View component, not the primary results View component.) This can significantly simplify the logic in the pages and make them easier to develop and maintain.

Controller components in Struts are Java classes and must be built using specific rules. They are usually referred to as "Action classes."

"Bottom Line Benefits of MVC"

Remember the earlier definition that described a Framework as "A set of assumptions, concepts, values, and practices that constitutes a way of viewing reality?" This is one of the most powerful benefits of the MVC pattern. It allows developers to think about (and design) complex applications as a series of relatively simple Model, View, and Controller components. This leads to better, more consistent, and more easily maintainable designs. In addition, it helps avoid the common pitfall of having each developer on a project choose a different approach for their work.

In other words, programmers wanted a framework that allowed them to focus on building the application without spending a time writing the code that organized and coordinated processing. After reviewing a number of other competing frameworks, they settled on Struts for a number of reasons—some technical and some not. The Struts technical features that were most imp:

1) Struts performs well.
2) Struts has a sound architectural model with a modest learning curve.
3) Struts is very solid and stable.
4) Struts has a strong set of custom tag libraries that simplify JSP development.

Also, according to programmer"Struts is very competitive technically, but to my mind the biggest advantages Struts has over competing technologies are practical." These features are:

1) ongoing development by a large number of committed users/developers
2) knowledgeable and responsive project leadership
3) generally quick (and sometimes near-instant) problem resolution via the Struts mailing list or archives
4) access to source code
5) strong connection to and commitment to future integration with forward-looking technologies like JSTL and Java Server Faces
6) increasing mind share that we expect will make it easier to hire new developers already familiar with the framework"

Conclusions

So, the question arises, should you consider adopting Struts? Of course, your answer depends on your particular circumstances and environment, but here are some criteria to consider:

Are you using the J2EE platform (that is, developing applications using J2EE-compliant servers such as Weblogic Server, Websphere Application Server, Jakarta Tomcat, JBoss, iPlanet, and so forth)? If the answer to this is yes, Struts is likely worth considering.

Do your developers have Java expertise? Although this isn't a "make or break" criterion, it helps. If your developers don't have Java experience but you are

dedicated to moving to Java anyway, Struts may actually make the transition easier.

Are you building applications that need to work in a Web browser? This is the niche that Struts fills. If the answer is no, Struts isn't for you. Unless you have requirements for browser-based application delivery, Struts won't add much value.

Are you building applications now that use JavaServer Pages (JSP)? If so, you should definitely be looking at Struts. It may increase your developer productivity significantly.

Is your development team considering building a "custom framework" for building Web-based applications? If so, ask them to justify not using Struts. Anything they would build on their own would likely not undergo near the amount of testing and development that Struts has. While they may have good reasons for not using it, Struts should definitely be on their radar.

To summarize, Struts is an application "Framework" for building Web-based applications in Java using the J2EE platform. Struts makes developers more productive by giving them prebuilt components to assemble applications from. Struts was built using industry best practices including the MVC design pattern and it can be deployed in a wide range of environments.

If you are using the Java/J2EE platform—and especially if you are currently developing applications using JavaServer Pages (JSP)—you should be considering Struts for the development of browser-based applications.

References:-

[1]. Holmes, J. (2006). *Struts: The Complete Reference, (Complete Reference Series)*. McGraw-Hill Osborne Media.

[2]. Franciscus, G., & McClanahan, C. R. (2002). *Struts in Action: Building web applications with the leading Java framework*. Manning Publications Co.

[3]. Shultz, A. M., Farha, O. K., Hupp, J. T., & Nguyen, S. T. (2009). A catalytically active, permanently microporous MOF with metalloporphyrin struts. *Journal of the American Chemical Society*, *131*(12), 4204-4205.

[4]. Holland, R., & Simpson, L. (1981). Finite-difference analysis of EMP coupling to thin struts and wires. *Electromagnetic Compatibility, IEEE Transactions on*, (2), 88-97.