Ejemplo 1 – Hola Struts

Como se ha aprendido en la arquitectura Struts 2, al hacer clic en un hipervínculo o envíe un formulario HTML en una aplicación web Struts 2, la entrada es recogida por el controlador que se envía a una clase Java llamados acciones. Después de que se ejecute la acción, un resultado selecciona un recurso para hacer la respuesta. El recurso es generalmente un JSP, pero también puede ser un archivo PDF, una hoja de cálculo Excel, o una ventana de applet de Java.

Usted asume ya la acumulación de su entorno de desarrollo. Ahora vamos a proceder a la construcción de nuestra primera **Hola Struts** proyecto struts2. El objetivo de este proyecto es construir una aplicación web que recoge el nombre del usuario y muestra "Hello World", seguido por el nombre de usuario.

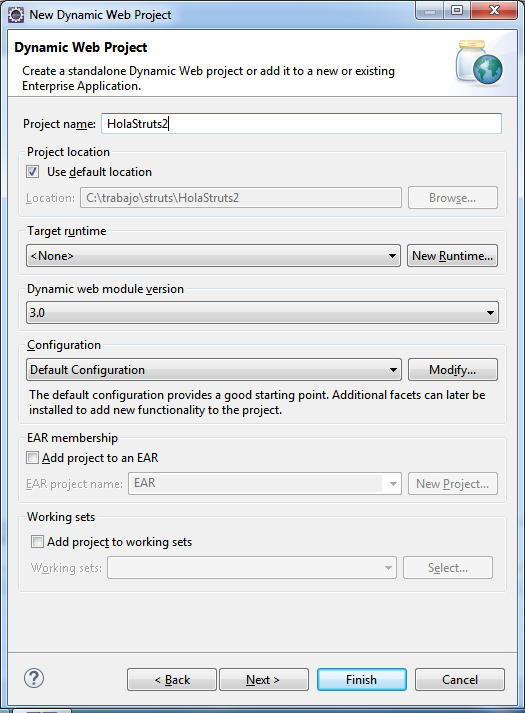
Tendríamos que crear después cuatro componentes para cualquier proyecto Struts 2:

|  |  |
| --- | --- |
| **SN** | **Componentes y descripción** |
| 1 | **Acción**  Crear una clase de acciones que contendrá la lógica de negocio completa y controla la interacción entre el usuario, el modelo, y la vista. |
| 2 | **Interceptores**  Crear interceptores si es necesario, o permite utilizar los interceptores existentes. Esto es parte del controlador. |
| 3 | **Vista**  Crear un JSP para interactuar con el usuario, toma de entrada de datos y presenta los mensajes finales. |
| 4 | **Archivos de configuración**  Crear archivos de configuración para acoplar la acción, vista y controladores. Estos archivos son struts.xml, web.xml, struts.properties. |

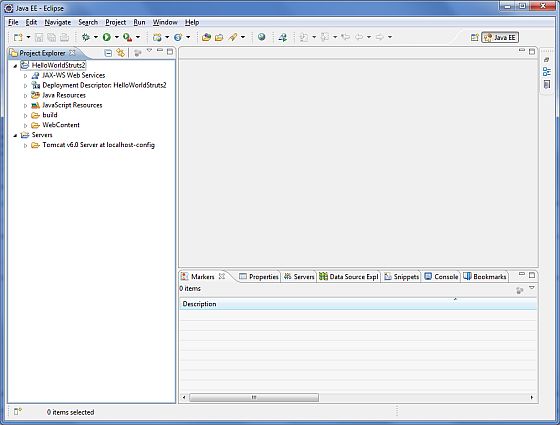
Vamos a utilizar Eclipse IDE, por lo que todos los componentes necesarios se crearán en virtud de un proyecto Web dinámico. Así que vamos a empezar con la creación dinámica de Project Web.

Crear un proyecto Web dinámico:

Comience el Eclipse y luego ir con **File> New> Web Dynamic Project** e introduzca el nombre del proyecto como **HolaStruts2** y establecer resto de las opciones como se indica en la siguiente pantalla:



Seleccione todas las opciones por defecto en las próximas pantallas y finalmente comprobar **Generar descriptor de despliegue Web.xml** opción. Esto creará un proyecto web dinámico para usted en Eclipse. Ahora vas con **Windows> Show View> Project Explorer** , y verá su proyecto ventana algo tan abajo:



Ahora copiar archivos siguientes de struts 2 lib carpeta **C:\struts-2.2.3\lib** al de nuestro proyecto **lib \WEB-INF\** carpeta. Para ello, sólo tiene que arrastrar y soltar todos los siguientes archivos en la carpeta WEB-INF \ lib.

* commons-fileupload-xyzjar
* commons-io-xyzjar
* commons-lang-xyjar
* commons-logging-xyzjar
* commons-logging-api-xyjar
* FreeMarker-xyzjar
* javassist-.xy.z.GA
* OGNL-xyzjar
* struts2-core-xyzjar
* XWork-core.xyzjar

Crear clase de acción:

Clase de acción es la clave para la aplicación Struts 2 y poner en práctica la mayor parte de la lógica de negocio en la clase de acción. Así que vamos a crear un archivo java HolaStrutsAction.java bajo **Java Resources > src** con un nombre de paquete **curso.struts2** con los contenidos que figuran a continuación.

La clase de acción responde a una acción del usuario cuando el usuario hace clic en un URL. Uno o más de los métodos de la clase de acción son ejecutados y un resultado de cadena se devuelve. Basándose en el valor del resultado, una página JSP específica se representa.

package curso.struts2;

public class HolaStrutsAction{

private String nombre;

public String execute() throws Exception {

return "success";

}

public String getNombre() {

return nombre;

}

public void setNombre(String nombre) {

this. nombre = nombre;

}

}

Esta es una clase muy simple con una propiedad llamada "name". Tenemos captadores estándar y métodos setter para la propiedad "name" y un método que devuelve la cadena "éxito" ejecutamos.

El marco Struts 2 creará un objeto de la clase HolaStrutsAction y llamar al método de ejecución en respuesta a la acción de un usuario. Usted pone su lógica de negocio dentro de ejecutar el método y finalmente devuelve la cadena constante. Simplemente decir para cada URL, usted tendría que aplicar una clase de acción y, o bien se puede utilizar ese nombre de la clase directamente como el nombre de la acción o puede asignar a algún otro nombre usando el archivo struts.xml como se muestra a continuación.

Crear una vista

Necesitamos un JSP para presentar el mensaje final, esta página será llamado por Struts 2 marco, cuando una acción predefinida va a suceder y esta asignación se define en el archivo struts.xml. Así que vamos a crear el archivo jsp continuación **HolaStruts.jsp** en la carpeta WebContent en su proyecto eclipse. Para ello, haga clic derecho en la carpeta WebContent en el explorador de proyecto y seleccione **Nuevo> Archivo JSP** .

<%@ page contentType="text/html; charset=UTF-8" %>

<%@ taglib prefix="s" uri="/struts-tags" %>

<html>

<head>

<title>Hola Struts</title>

</head>

<body>

Hola Struts, <s:property value="nombre"/>

</body>

</html>

La directiva taglib dice el contenedor de servlets que esta página va a utilizar las etiquetas de Struts 2 y que estas etiquetas será precedido por s. El s: Etiqueta propiedad muestra el valor de la propiedad de clase de acción nombre "> que se devuelve mediante el método **getNombre ()** de la clase HolaStrutsAction.

Crear página principal:

También tenemos que crear **index.jsp** en la carpeta WebContent. Este archivo servirá como la URL acción inicial donde un usuario puede hacer clic para decir a los Struts 2 marco para llamar al método definido de la clase HolaStrutsAction y hacer la vista HolaStruts.jsp.

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<%@ taglib prefix="s" uri="/struts-tags"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"

"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<title>Hola Struts</title>

</head>

<body>

<h1>Hola Mundo desde Struts2</h1>

<form action="saludar">

<label for="nombre">Por favor introduce tu nombre:</label><br/>

<input type="text" name="nombre"/>

<input type="submit" value="SALUDAR"/>

</form>

</body>

</html>

El **saludar** acción definida en el archivo de vista anterior se asigna a la clase HolaStrutsAction y su método de ejecución usando el archivo struts.xml. Cuando un usuario hace clic en el botón Enviar que hará que el FrameWork Struts 2 para ejecutar el método de ejecución definido en la clase HolaStrutsAction y con base en el valor devuelto del método, se seleccionará a una conclusión acertada y representa como una respuesta.

Archivos de configuración

Necesitamos un mapeo para capturar la URL, la clase HolaStrutsAction (Modelo), y el HolaStruts.jsp (la vista )están juntos. El mapeo indica al FrameWork qué clase responderá a la acción del usuario (la URL), que se ejecutará el método de esa clase, y lo que a fin de render basado en la cadena de resultados que método devuelve.

Así que vamos a crear un archivo llamado **struts.xml**.  Struts 2 requiere que struts.xml esté presente en la carpeta de clases. Así que crear el archivo struts.xml bajo la carpeta WebContent/WEB-INF/classes. Eclipse no crea la carpeta "clases" por defecto, por lo que necesita para hacerlo usted mismo. Para ello, haga clic derecho en la carpeta WEB-INF en el explorador de proyecto y seleccione **New> Folder**. Su struts.xml debe ser similar a:

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE struts PUBLIC

"-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"

"http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>

<constant name="struts.devMode" value="true" />

<package name="holaStruts" extends="struts-default">

<action name="saludar"

class="curso.struts2.HolaStrutsAction"

method="execute">

<result name="success">/HolaStruts.jsp</result>

</action>

</package>

</struts>

Pocas palabras sobre el archivo de configuración anterior. Aquí establecemos la constante **struts.devMode** a true, porque estamos trabajando en el entorno de desarrollo y tenemos que ver algunos mensajes de registro útiles. Entonces, definimos un paquete llamado **holaStruts**. La creación de un paquete es útil cuando se desea agrupar sus acciones en conjunto. En nuestro ejemplo, hemos llamado a nuestra acción como "saludar", que se corresponde con la URL **/saludar.action** y está respaldado por el **HolaStrutsAction.class**. La **ejecución** del método **HolaStrutsAction.class** es el método que se ejecuta cuando se invoca la URL**/saludar.action** . Si el resultado de la **ejecución de** método devuelve el "éxito", y luego tomamos el usuario **HolaStruts.jsp** .

El siguiente paso es crear un **web.xml** archivo que es un punto de entrada para cualquier solicitud de Struts 2. El punto de entrada de la solicitud Struts2 será un filtro definido en el descriptor de despliegue (web.xml). Por lo tanto vamos a definir una entrada oforg.apache.struts2.dispatcher.FilterDispatcher clase en web.xml. El archivo web.xml se debe crear en la carpeta WEB-INF bajo WebContent. Eclipse ya había creado un archivo web.xml esqueleto para usted al crear el proyecto. Por lo tanto, sólo permite modificarlo de la siguiente manera:

<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns="http://java.sun.com/xml/ns/javaee"

xmlns:web="http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd"

xsi:schemaLocation="http://java.sun.com/xml/ns/javaee

http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd"

id="WebApp\_ID" version="3.0">

<display-name>Struts 2</display-name>

<welcome-file-list>

<welcome-file>index.jsp</welcome-file>

</welcome-file-list>

<filter>

<filter-name>struts2</filter-name>

<filter-class>

org.apache.struts2.dispatcher.FilterDispatcher

</filter-class>

</filter>

<filter-mapping>

<filter-name>struts2</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

</web-app>

Hemos especificado index.jsp para ser nuestro archivo de bienvenida. Luego hemos configurado el filtro Struts2 para correr en todas las direcciones URL (es decir, cualquier URL que coinciden con el patrón **/ \***)

Habilitar registro detallado:

Puede habilitada la funcionalidad completa de registro mientras trabaja con Struts 2 creando **logging.properties** archivo bajo **WEB-INF/classes** carpeta. Mantenga las dos líneas siguientes en el archivo de propiedades:

org.apache.catalina.core.ContainerBase.[Catalina].level = INFO

org.apache.catalina.core.ContainerBase.[Catalina].handlers =

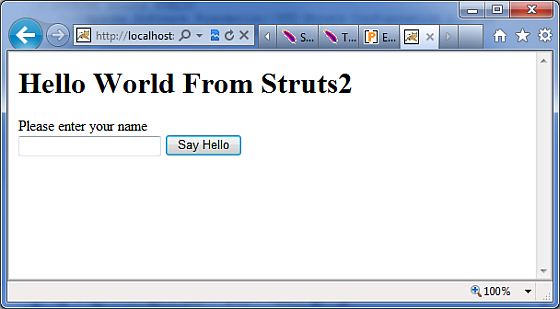
\java.util.logging.ConsoleHandler

Los logging.properties defecto especifica una ConsoleHandler para encaminar el registro a la salida estándar y también un FileHandler. Umbral de nivel de registro de un controlador se puede ajustar con GRAVE, ADVERTENCIA, INFO, CONFIG, FINE, FINER, FINEST o ALL.

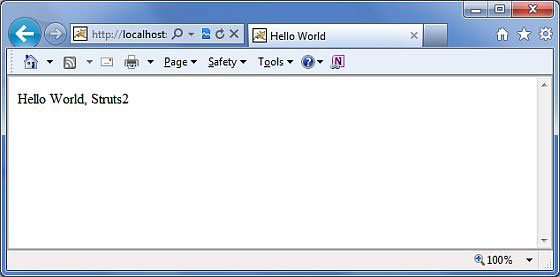
Eso es todo. Estamos dispuestos a ejecutar nuestra aplicación HolaStruts usando Struts 2.

Ejecutar la Aplicación

Haga clic derecho sobre el nombre del proyecto y haga clic en **Exportar> WAR File** para crear un archivo de WAR. Entonces implementar esta WAR en el directorio webapps del Tomcat. Por último, iniciar el servidor Tomcat y tratar de acceso URL http://localhost:8080 /HolaStruts2/index.jsp. Esto le dará la siguiente pantalla:



Introduzca un valor "Struts2" y presentar la página. Usted debe ver la siguiente página



Tenga en cuenta que puede definir **el índice** como una acción en el archivo struts.xml y en ese caso usted puede llamar a la página índice como http: // localhost: 8080 / HelloWorldStruts2 / index.action. Revise a continuación cómo se puede definir como un índice de la acción:

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE struts PUBLIC

"-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"

"http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>

<constant name="struts.devMode" value="true" />

<package name="holaStruts" extends="struts-default">

<action name="index">

<result >/index.jsp</result>

</action>

<action name="hello"

class="curso.struts2.HolaStrutsAction"

method="execute">

<result name="success">/HolaStruts.jsp</result>

</action>

</package>

</struts>