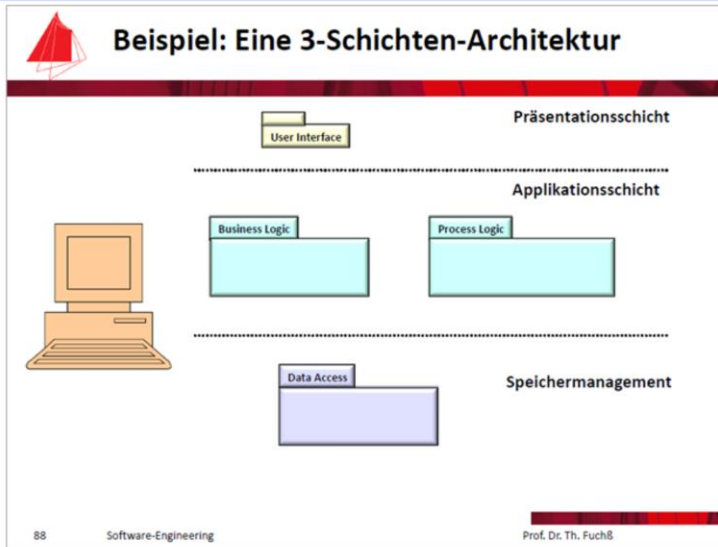


- Vorstellung des Frameworks Struts 2

Labor für Verteilte Systeme
Master
Fachgebiet Informatik
Hochschule Karlsruhe

Dipl.-Inform.(FH) Adelheid Knodel

3-Schichten-Architektur



Vorstellung Struts2 SS 2020

2

Adelheid Knodel

Um Software Anwendungen zu strukturieren gibt es verschiedene Architekturmuster. Der Webshop ist in der 3 Schichten Architektur realisiert.

3 Schichten Architektur heißt, es gibt die Präsentationsschicht, die Anwendungsschicht und die Persistenzschicht.

Die Präsentationsschicht beinhaltet die Komponenten für das User Interface, d.h. für die Darstellung der Information für den Benutzer und die Komponenten für die Eingaben des Users.

Die Anwendungsschicht enthält die Businesslogik und die Ablauflogik. D.h. hier werden die Daten verarbeitet und die Anfragen des Users bearbeitet und anhand der Ergebnisse der Verarbeitung die Darstellung auf der Präsentationsschicht veranlasst.

Die dritte Schicht ist die Persistenzschicht, also die Schicht, die Objekte persistiert, d.h. Objekte permanent auf ein Medium speichert und von dort auch wieder liest. Dies kann eine Datenbank sein, oder Dateien im Filesystem oder andere Möglichkeiten zur Speicherung von Daten.

Als weiteres Muster kommt im Webshop das MVC Pattern zum Einsatz. Ein Framework welches dieses Muster umsetzt, ist das Struts2 Framework. Dieses Framework wird im Webshop eingesetzt.

Struts2 Inhalt

- MVC Design
- MVC Komponenten
- Funktionsweise
- Komponenten im Webshop

Inhalt dieser Folien ist ein kurzer Überblick über dieses Framework.

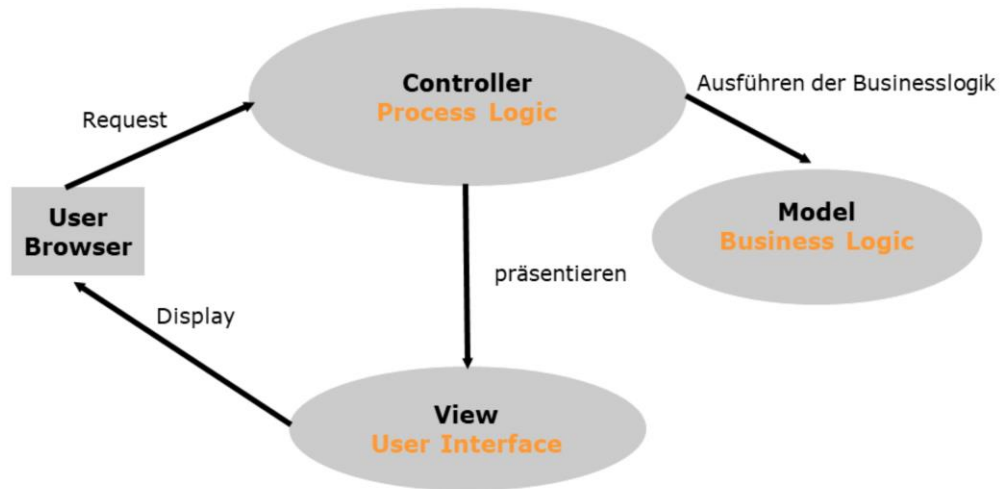
Was bedeutet prinzipiell MVC Design

Was sind die Komponenten in Struts2, die dieses Muster umsetzen.

Wie funktioniert der Ablauf, wie arbeiten die einzelnen Komponenten zusammen.

Und zum Schluss noch, wo finden Sie die einzelnen Komponenten in der Verzeichnisstruktur des Webshops

MVC Architektur



Vorstellung Struts2 SS 2020

4

Adelheid Knodel

Ein Weg Zuständigkeiten einer Software zu trennen ist der Einsatz des Model-View-Controller Musters.

Das Model steht für den Code der Businesslogik, die View repräsentiert die Darstellung der Daten für den User und der Controller steht für die Ablauflogik.

Das Struts2 Framework stellt den Rahmen zur Verfügung, um diese MVC Architektur in Webapplikationen leichter umsetzen zu können.

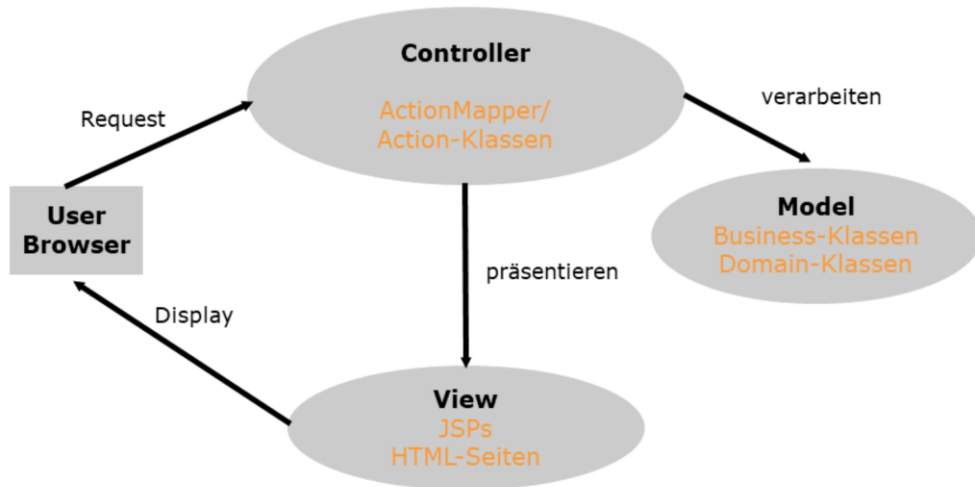
Vorteile des MVC Designs

- Änderung der Funktionalität ohne Änderung der Seiten

- Änderung der Seiten ohne Änderung der Logik

- Bessere Strukturierung und damit Überschaubarkeit

MVC Komponenten in Struts2



Vorstellung Struts2 SS 2020


5

Adelheid Knodel

Welches sind die Komponenten in Struts2, die das MVC Muster umsetzen.

Die Komponenten, die in Struts2 die die MVC Komponenten umsetzen, sind:

- der Controller wird durch die sogenannten Actionklassen und den Actionmapper implementiert
- das Model durch die Businessklassen oder Domainklassen, die die Daten verarbeiten
- die View wird durch die Java Server Pages, HTML Seiten oder andere Darstellungsmöglichkeiten repräsentiert



Hochschule Karlsruhe
Technik und Wirtschaft
UNIVERSITY OF APPLIED SCIENCES

Beispiel View: JSP/ Browser

VS Labor Webshop mit Bootstrap 3

http://localhost:8080/WebShopStart/

Informatik

VS Lab EShop

Login

Username*:

Password*

Noch nicht registriert?

login

© Copyright 2016 Informatik - Hochschule Karlsruhe

Vorstellung Struts2 SS 2020

6

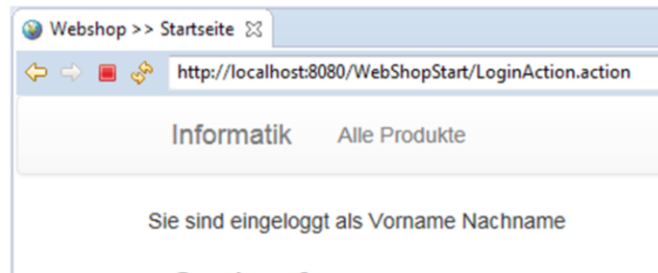
Adelheid Knodel

Am Beispiel des zu analysierenden Webshops wird kurz der Ablauf erläutert:

Nach Eingabe der URL: localhost:8080/WebShopStart

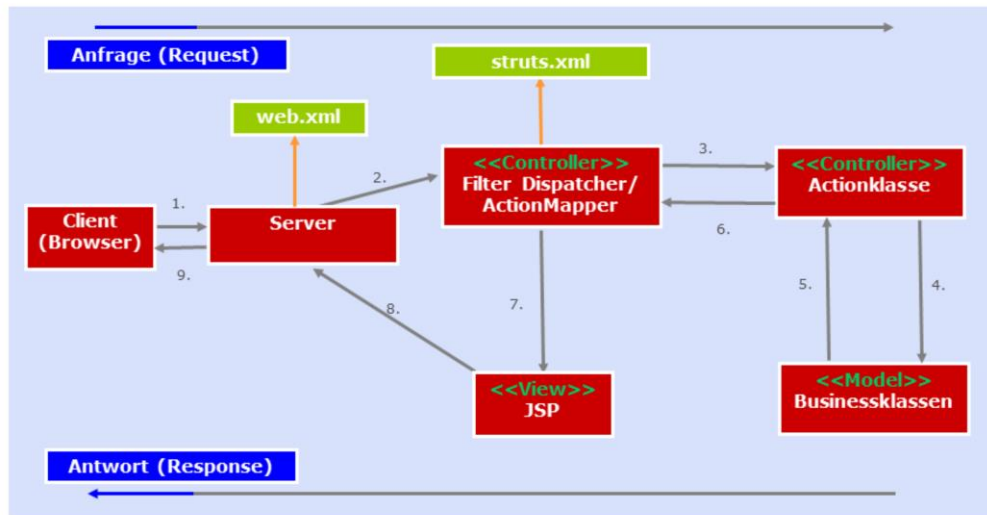
Erscheint die Startseite des Webshops, auf der Startseite können die Zugangsdaten eingegeben werden,
mit dem Klick auf den Button „login“ wird ein Login Request an den Server gesendet,
dort wird die Anfrage bearbeitet,
bei korrekten Zugangsdaten wird die Webseite auf der nächsten Folie erzeugt und an den User zurückgesendet

Beispiel View: JSP /Browser



bei korrekten Zugangsdaten wird angezeigt, dass der Benutzer erfolgreich angemeldet werden konnte

Funktionsprinzip und Ablauf eines Requests



Vorstellung Struts2 SS 2020

8

Adelheid Knodel

Die Anfrage und das Antworten läuft folgendermaßen ab:

1. Der Benutzer sieht eine Webseite, kann dort Daten in ein Formular eingeben, anschließend wird durch das Klicken auf einen Button der Request an den Server geschickt,
2. Der Server ist über die web.xml so konfiguriert, dass er weiß, was mit diesem Request zu tun ist. In diesem Fall ist er so konfiguriert, dass die Requests über die StandardFilterChain des Servlet Containers an den Filterdispatcher gesendet werden, der dann den Actionmapper aufruft.
3. Der Actionmapper hat für jede Action, die durch z.B. einen Button ausgelöst werden kann, eine zugeordnete Actionklasse konfiguriert, die Konfiguration ist in der struts.xml festgelegt
4. Die Actionklasse ruft die entsprechenden Businessklassen auf, um die Anforderung zu verarbeiten
5. Die Businessklasse gibt das Ergebnis zurück an die Actionklasse
6. Die Actionklasse erzeugt anhand des Ergebnisses aus den Businessklassen, das Ergebnis, das an den ActionMapper zurückgegeben wird
7. Der ActionMapper muss jetzt wieder in der Konfiguration struts.xml ermitteln, welche Seite anhand des Ergebnisses erzeugt werden soll
8. und an den Servlet Container zurückgegeben wird
9. Der Servlet Container sendet als Antwort die erzeugte Seite an den Browser zurück

Struts2 Controller Komponenten

- Hauptkomponente
`org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter`
`org.apache.struts2.dispatcher.mapper.ActionMapper`
- Konfiguration des Filters in `web.xml` des Projekts
- Definition des `ActionMapping` in `struts.xml`

Die zentralen Komponenten von Struts sind die Komponenten für den Controller

- Der `StrutsPrepareAndExecuteFilter` und
- Der `ActionMapper`

Als auch die `web.xml` und die `struts.xml`, in der `web.xml` wird der `StrutsPrepareAndExecuteFilter` konfiguriert,

In der `struts.xml` werden die Actions festgelegt und wie sie verarbeitet werden sollen.

web.xml des Struts Controller konfigurieren

web.xml Deployment Descriptor File, beschreibt wie eine Web-Applikation in einem Servlet Container, wie z.B. Tomcat, konfiguriert werden soll.

```
<!-- Filter Dispatcher Configuration -->
<filter>
  <filter-name>struts2</filter-name>
  <filter-class>
    org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter
  </filter-class>
</filter>

<!-- Filter Mapping -->
<filter-mapping>
  <filter-name>struts2</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

Die web.xml beschreibt wie eine Webanwendung HTTP Requests durch den Servlet Container bearbeitet, für eine Struts2 Anwendung muss dort die Klasse für den Filterdispatcher konfiguriert sein.

In der dargestellten Datei wird das Mapping für eingehende Requests so festgelegt:

Das FilterMapping legt fest, dass alle eingehenden Requests **<url-pattern>/*</url-pattern>** auf den Filter mit dem Namen **struts2** gemappt werden.

Dieser Filter **struts2** ist in der Filter Dispatcher Configuration auf die Klasse **org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter** gemappt

Dieses File web.xml muss im Webprojekt im Verzeichnis **WebContent/WEB-INF** stehen.

struts.xml

struts.xml enthält das Action Mapping

```
<struts>
  ...
  <!-- Action Definitions -->
  <package name= "package1" extends="struts-default">
    <action name="action1">
      .
    </action>
    .
    <action name="actionnn">
      .
    </action>
  </package>
  .
  <package name= "packageM" extends="struts-default">
    </package>
  ...
</struts>
```

Die struts.xml enthält das Action Mapping und ist folgendermaßen aufgebaut:

Die Definitionen können in verschiedene Packages aufgeteilt werden,
Jedes Package enthält mehrere Action Definitionen

struts.xml

struts.xml ist einer der zentralen Punkte des Struts2 Frameworks

```
<action name="action1"  
    class = "package1.subpackage1.ActionClass1" method="execute">  
    <result name="success">mypage.jsp</result>  
    <result name="input"> input.jsp</result>  
    <result name="result1">myresultpage1.jsp</result>  
</action>
```

execute method = default

Für jede Action wird dann definiert :
ein Name und eine Klasse, die die Anforderung ausführt und
mindestens ein Result Name und eine dazugehörige Seite

Beispiel action in index.jsp

```
<%@ page contentType="text/html; charset=UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<%@ taglib prefix="s" uri="/struts-tags" %>

<html>
  <body>
```

```
    <s:form action="LoginAction" focusElement="username">

      <s:textfield name="username" />

      <s:password name="password" /><br>

      <s:submit method="execute" value="login" align="center"/>
    </s:form>
  </body>
</html>
```

Vorstellung Struts2 SS 2020

13

Adelheid Knodel

Wie sieht das Ganze konkret aus:

Die dargestellte Login Seite sieht als Jsp Code so aus,

es gibt das Form Tag **s:form**,

In diesem Tag wird der Action Name (**LoginAction**) festgelegt

Es gibt 2 Felder: Username (*s:textfield name="username"*) und Passwort (*s:password name="password"*)

Und den Button Login (*s:submit method="execute" value="login"*)

Ablauf

- Properties aus JSP werden zu Request-Parametern
`<s:textfield name="username"/>`
`<s:password name="password"/>`
- Request 'LoginAction' mit Parametern wird an Server gesendet
`<s:form action= "LoginAction"`

Username:

Passwort *

Noch nicht registriert?

Status	Methode	Datei	Host
200	POST	LoginAction.action?sessionId=7CAA07...	localhost:8080
404	GET	custom.css	localhost:8080

Kopfzeilen

Anfrageparameter durchsuchen

Formulardaten

method: execute: login
password: Test
username: Test

Vorstellung Struts2 SS 2020

14

Adelheid Knodel

Über den Button Login wird ein Request LoginAction ausgelöst,
 Die Felder im Formular werden zu Request Parametern,
 die Eingaben in den Feldern werden zu Werten für die Requestparameter

Ausschnitt struts.xml

(...)

```
<package name="vislabExample" extends="struts-default">
  <action name="LoginAction"
    class="hska.iwi.eShopMaster.controller.LoginAction"
    <result name="input"> index.jsp </result>
    <result name="success"> /pages/welcome.jsp</result>
  </action>
</package>
```

(...)

in Login.jsp
<s:form action="LoginAction">

In der struts.xml ist die Action mit dem Name LoginAction konfiguriert und auf die Klasse `hska.iwi.eShopMaster.controller.LoginAction` gemappt

Es ist darauf zu achten, dass die Schreibweise in der JSP mit der Schreibweise in der struts.xml übereinstimmen muss.

Ausschnitt struts.xml

(...)

```
<package name="vislabExample" extends="struts-default">
  <action name="LoginAction"
    class="hska.iwi.eShopMaster.controller.LoginAction"
    method="execute">
    <result name="input">index.jsp</result>
    <result name="success"> /pages/start.jsp</result>
  </action>
</package>
```

(...)

Die Methode "execute" ist die default-Methode und kann deshalb auch weggelassen werden.

Die konfigurierte Klasse LoginAction muss eins der beiden Resultate liefern,
entweder
input oder success,
Diese Namen können frei gewählt werden.

Ablauf

- Properties aus JSP werden Request-Parameter
- Request LoginAction an Server
- Server entscheidet anhand der web.xml:
alle Anfragen, werden an den Struts2 FilterDispatcher weitergeleitet
- FilterDispatcher entscheidet anhand der struts.xml welche
ActionKlasse aufgerufen wird, die Action Klasse bearbeitet
die Eingabe, produziert Ergebnisse
- FilterDispatcher entscheidet anhand der Ergebnisse und der
Definitionen in der struts.xml welche **View** (Seite)
ausgegeben wird

Hier noch mal zusammengefasst der Ablauf eines Requests

Struts Action Klasse

Action Klassen sind ein zentraler Punkt des Struts2 Frameworks

- Jeder Request/URL ist auf eine Methode einer Action gemappt
- Transfer der Daten von und zur View
- Validierung von Eingabedaten
- Aufruf der Methoden zur Verarbeitung der Anforderung (Businesslogik)
- Welche Seite soll als Ergebnis dargestellt werden

Struts Action Klasse

- muss eine execute - Methode definieren
- muss für jedes Eingabe-/ Ausgabefeld eine Variable (Property) mit Get-/Set-Methode definieren
Struts wandelt HTTP Parameter in JavaBean Properties und füllt HTML Felder aus JavaBean Properties
- kapselt die Verarbeitung der Daten für einen Request
- Muss als Return-Wert einen String zurückgeben

Beispiel property in index.jsp

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

<html>
  <body>

    <s:form action="LoginAction" focusElement="username">
      <s:textfield name="username" />
      <s:password name="password" /> <br>

      <s:submit method="execute" value="login" align="center"/>
    </s:form>
  </body>
</html>
```

In dieser JSP Seite werden zwei Felder username und passwort festgelegt, Um verarbeitet werden zu können, müssen diese Felder auch in der Actionklasse wieder erscheinen.

Action Class Beispiel

```
public class LoginAction extends ActionSupport {  
  
    private String username;    // getter und setter fehlen aus Platzgründen  
    private String password;    // getter und setter fehlen aus Platzgründen  
  
    public String execute() throws Exception {  
  
        if (getUsername().equals("user")) {  
            if (getPassword().equals("password")) {  
                **  
                return SUCCESS;  
            } else {  
                addActionError(getText("error.user.passwordforgotten"));  
                addActionError("Bitte geben Sie das richtige Passwort ein!");  
                return "input";  
            }  
        }  
        else { addActionError(getText("error.username.register"));  
            return INPUT;  
        }  
    }  
}
```

In der Aktionklasse müssen für alle zu verarbeitenden Properties entsprechende Variablen mit Getter und/oder Setter Methoden existieren.

Der Automatismus von Struts2 erwartet zum Schreiben der Daten bzw. Requestparameter in die Variablen der Actionklasse setter Methoden, zum Lesen bzw. zum Füllen der Daten in die Ergebnisseite entsprechende Getter-Methoden

Action Class Beispiel

```
public class LoginAction extends ActionSupport {

    private String username;           // getter und setter fehlen aus Platzgründen
    private String password;          // getter und setter fehlen aus Platzgründen

    public String execute() throws Exception {

        if (getUsername().equals("user") {
            if (getPassword().equals("password")) {
                ..
                return SUCCESS;
            } else {
                addActionError(getText("error.user.passwordforgotten"));
                addActionError("Bitte geben Sie das richtige Passwort ein!");
                return "input";
            }
        } else { addActionError(getText("error.username.register"));
                return INPUT;
            }
        }
    }
}
```

Als Ergebnis, also den Rückgabewert der Methode erwartet Struts 2 einen String,

Der String kann beliebig gewählt werden. Es gibt vordefinierte Konstante wie SUCCESS, INPUT, ERROR

“input“ und INPUT sind gleichbedeutend

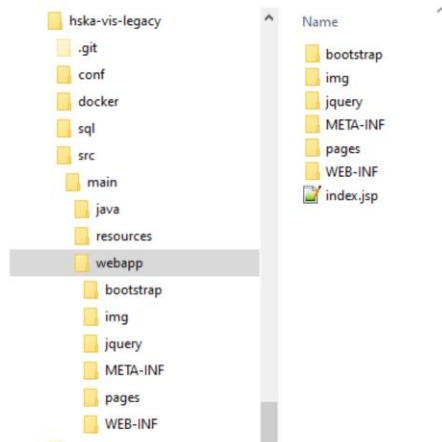
Ausschnitt struts.xml

```
<action name="LoginAction"
        class="hska.iwi.eShopMaster.controller.LoginAction">
    <result name="input">index.jsp</result>
    <result name="success"> /pages/start.jsp</result>
</action>
```

Verzeichnisstruktur des Projekts

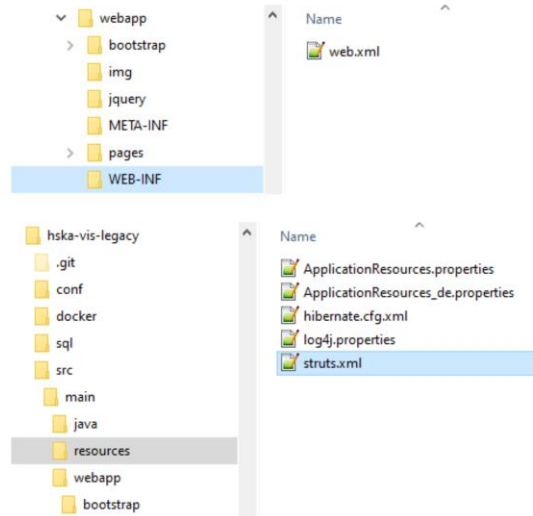
Apache Tomcat ist ein Open Source Webserver und Webcontainer, mit dem in Java geschriebene Web-Anwendungen auf Servlet- bzw. JSP-Basis ausgeführt werden können.

Verzeichnisstruktur der geklonten Web-Anwendung



Struts Konfigurationsfiles

web.xml



struts.xml

Links

Struts:

<https://struts.apache.org/index.html>

http://www.tutorialspoint.com/struts_2/index.htm

<http://www.roseindia.net/struts/index.shtml>

<http://www.javajazzup.com/issue5/page31.shtml>

<https://cwiki.apache.org/confluence/display/WW/Tag+Reference>

Projekt clonen von

<https://github.com/mavogel/hska-vis-legacy/>