AJAX och A4J

AJAX

- Asyncronos Javascript and XML
- Ett sätt att få en annan typ av kommunikation mellan klient och server
- Istället för att vi för över sida för sida så ska man föra över data.
- när en komponent på våran sida behöver uppdataeras så uppdaterar man bara den komponenten inte hela sidan.

- AJAX är inget hokuspokus.
- De ramverk som finns är helt enkelt javascript kod som är inkapslat i lite snygga ramverk
- AJAX bygger kring detta objekt (ActiveXObject i InternetExplorer)

Litet exempel

- <html>
- <body>
- <script type="text/javascript">
- function ajaxFunction() {
- var xmlHttp;
- xmlHttp=new XMLHttpRequest();
- xmlHttp.onreadystatechange=function()
- {
-

```
    if(xmlHttp.readyState==4)
        {
        document.myForm.time.value=xmlHttp.responseText;
        }
      }
    xmlHttp.open("GET","time.asp",true);
    xmlHttp.send(null);
    }
</script>
```

- <form name="myForm">
- Name: <input type="text"
- onkeyup="ajaxFunction();" name="username" />
- Time: <input type="text" name="time" />
- </form>
-
- </body>
- </html>

Ajax4JSF

- Är ett opensource ramverk som adderar lite javascript funktionallitet till JSF.
- Finns även ett Ajax4JSF rich komponenter .
- Förändrar ingen av JSFs funktionallitet utan är ett tillägg.
- Allt du behöver är en Servlet/JSP container och lite jar filer från a4j.

- aj4 lägger till javascript funktionallitet utan att du behöver skriva den, YES!!!
- Vanliga taggar som JSF ungefär.
- Du lägger till ett taglib i JSP filen
- två jar filer måste till i WEB-INF/lib
  - ajax4jsf-1.1.1.jar
  - oscache-2.3.jar
- a4j laddas ner från http://labs.jboss.com/jbossajax4jsf/

- En request från JSP sidan går till AJAX motorn på clienten
- Om det är en submit så skickas data till servern som XML anars updaterar den direkt sidan via javascript.
- På server sidan packas XML upp och förvandlar detta till en JSP request som utförs som vanligt.
- Resultatet tolkas och transformeras till XML som sedan skickas tillbaka till Ajax motorn på klientsidan som updaterar sidan.

Vad den kan trigga

- I princip kan den trigga på allt som javascripts events.
- onkeyup, onclick, onchange, onmouseover, onfocus, onsubmit, onmouseout

Taggar

- <a4j:support>
  - Viktiga attribut
  - reRender, id på komponent som den ska om rendrera.
  - rendered, om false så rendreras inte komponenten.
  - event, namn på händelsen som den ska trigga på
  - requestDelay, väntan mellan request som skickas.
- <h:/inputText value=”#{backbean.tjo}” >
- <a4j:support event=”onmouseover” reRender=”somekomponent”>
- </h:/inputText>

- <a4j:poll>
  - Används för att polla servern och kolla om data updateras.
  - <a4j:poll intervall=”1000” reRender=”enkomponent”>
-

- <a4j:outputPanel>
  - Används för att grupperaihop komponenter.
  - Används mest då man vill använda reRender och vill peka på en större grupp.

- <a4j:mediaOutput>
  - Används främst för att generera dynamiska bilder som kan uppdateras on the fly!
  - <a4j:mediaOutput element=img cacheable=false session=true createContent=”#{paintBean.paint}” value=”#{paintData}” mimeType=”image/jpeg”>
  - På servern i klassen PaintData
  - public void paint(OutputStream out, Object data)

- <a4j:log>
  - Debug log för javascripten.
- <a4j:htmlCommandLink>, <a4jCommandLink>
  - Nästan samma som JSF, men gömda variabler som parametrar är genererade i förväg.
- <a4j:form>
  - Nästan samma som JSF, men gömda variabler är genererade

Exempel

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%@taglib uri="https://ajax4jsf.dev.java.net/ajax" prefix="a4j"%>

<%@taglib prefix="f" uri="http://java.sun.com/jsf/core"%>

<%@taglib prefix="h" uri="http://java.sun.com/jsf/html"%>

<html>

    <body>

        <f:view>
```

```
        <h:messages/>
          <h:form>
              <h:inputText id="tjo" size="30" value="#{bean.text}" >
                  <a4j:support event="onkeyup" reRender="rep" />
              </h:inputText>
              <h:outputText value="#{bean.text}" id="rep" />
          </h:form>
      </f:view>
    </body>
</html>
```

TestBean.java

```
package test;

public class TestBean {

    private String str;

    public String getText() {

        return str;

    }

    public void setText(String str) {

        this.str = str;

    }

   }
```

web.xml

- <?xml version="1.0" encoding="UTF-8"?>
- <web-app version="2.4" xmlns="http://java.sun.com/xml/ns/j2ee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">
- <filter>
- <display-name>Ajax4jsf Filter</display-name>
- <filter-name>ajax4jsf</filter-name>
- <filter-class>org.ajax4jsf.Filter</filter-class>
- </filter>
- <filter-mapping>

```
<filter-name>ajax4jsf</filter-name>
    <servlet-name>Faces Servlet</servlet-name>
    <dispatcher>REQUEST</dispatcher>
    <dispatcher>FORWARD</dispatcher>
    <dispatcher>INCLUDE</dispatcher>
</filter-mapping>
<context-param>
    <param-name>com.sun.faces.verifyObjects</param-name>
    <param-value>false</param-value>
</context-param>
<context-param>
```

- <param-name>com.sun.faces.validateXml</param-name>
- <param-value>true</param-value>
- </context-param>
- <context-param>
- <param-name>javax.faces.STATE_SAVING_METHOD</param-name>
- <param-value>client</param-value>
- </context-param>
- <servlet>
- <servlet-name>Faces Servlet</servlet-name>
- <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
- <load-on-startup>1</load-on-startup>
- </servlet>
- <servlet-mapping>

```
    <servlet-name>Faces Servlet</servlet-name>
  <url-pattern>/faces/*</url-pattern>
</servlet-mapping>
<session-config>
  <session-timeout>
    30
  </session-timeout>
</session-config>
<welcome-file-list>
  <welcome-file>
    index.jsp
  </welcome-file>
</welcome-file-list>
```

faces-config.xml

```
<?xml version='1.0' encoding='UTF-8'?>
<faces-config>
    <managed-bean>
        <managed-bean-name>bean</managed-bean-name>
        <managed-bean-class>test.TestBean</managed-bean-class>
        <managed-bean-scope>session</managed-bean-scope>
        <managed-property>
            <property-name>text</property-name>
            <value>bajen!</value>
        </managed-property>
    </managed-bean>
</faces-config>
```