



# JSF: The Ajax4jsf Library

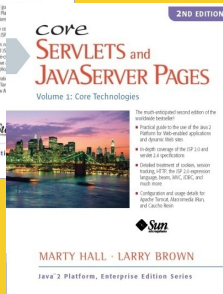
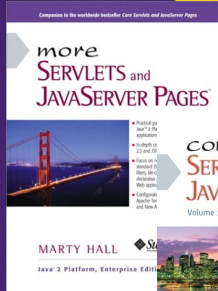
Originals of Slides and Source Code for Examples:

<http://www.coreservlets.com/JSF-Tutorial/>

This somewhat old tutorial covers JSF 1, and is left online for those maintaining existing projects. All new projects should use JSF 2, which is both simpler and more powerful. See <http://www.coreservlets.com/JSF-Tutorial/jsf2/>.

**Customized Java EE Training:** <http://courses.coreservlets.com/>

Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Hadoop, Android. Developed and taught by well-known author and developer. At public venues or onsite at *your* location.



**For live training on JSF 1 or 2, please see courses at <http://courses.coreservlets.com/>.**



**Taught by the author of *Core Servlets and JSP*, *More Servlets and JSP*, and this tutorial. Available at public venues, or customized versions can be held on-site at your organization.**

- Courses developed and taught by Marty Hall
    - JSF 2, PrimeFaces, servlets/JSP, Ajax, jQuery, Android development, Java 6 or 7 programming, custom mix of topics
    - Ajax courses can concentrate on 1 library (jQuery, Prototype/Scriptaculous, Ext-JS, Dojo, etc.) or survey several
  - Courses developed and taught by [coreservlets.com](http://coreservlets.com) experts (edited by Marty)
    - Spring, Hibernate/JPA, EJB3, GWT, Hadoop, SOAP-based and RESTful Web Services
- Contact [hall@coreservlets.com](mailto:hall@coreservlets.com) for details**

# Topics in This Section

- **Ajax motivation**
- **Installation**
- **Main Ajax4jsf Elements**
  - a4j:commandButton
  - a4j:commandLink
  - a4j:poll
  - a4j:support
- **Limitations on use of h:outputText with Ajax4jsf**
- **Other Ajax Toolkits**

5

© 2012 Marty Hall



## Motivation

**Customized Java EE Training: <http://courses.coreservlets.com/>**

Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Hadoop, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

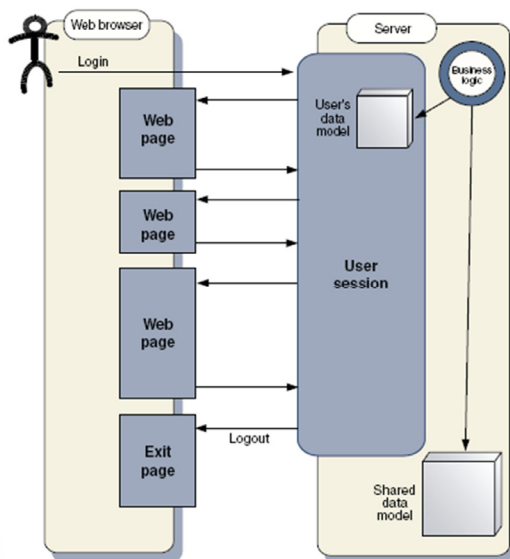
# Why Ajax?

- **HTML and HTTP are weak**
  - Non-interactive
  - Coarse-grained updates
- **Everyone wants to use a browser**
  - Not an application that requires SW installation
- **“Real” browser-based active content**
  - Failed: Java Applets
    - Not universally supported; can't interact with the HTML
  - Serious alternative: Flash (and Flex)
    - Not yet universally supported; limited power
  - New and unproven
    - Microsoft Silverlight
    - JavaFX
    - Adobe AIR (formerly “Apollo”)

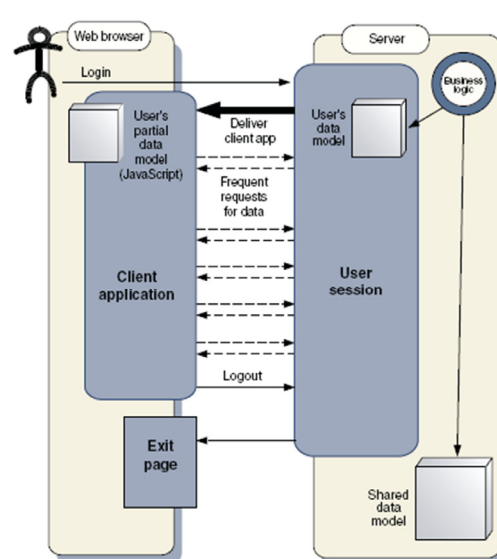
7

# Traditional Web Apps vs. Ajax Apps

Infrequent large updates



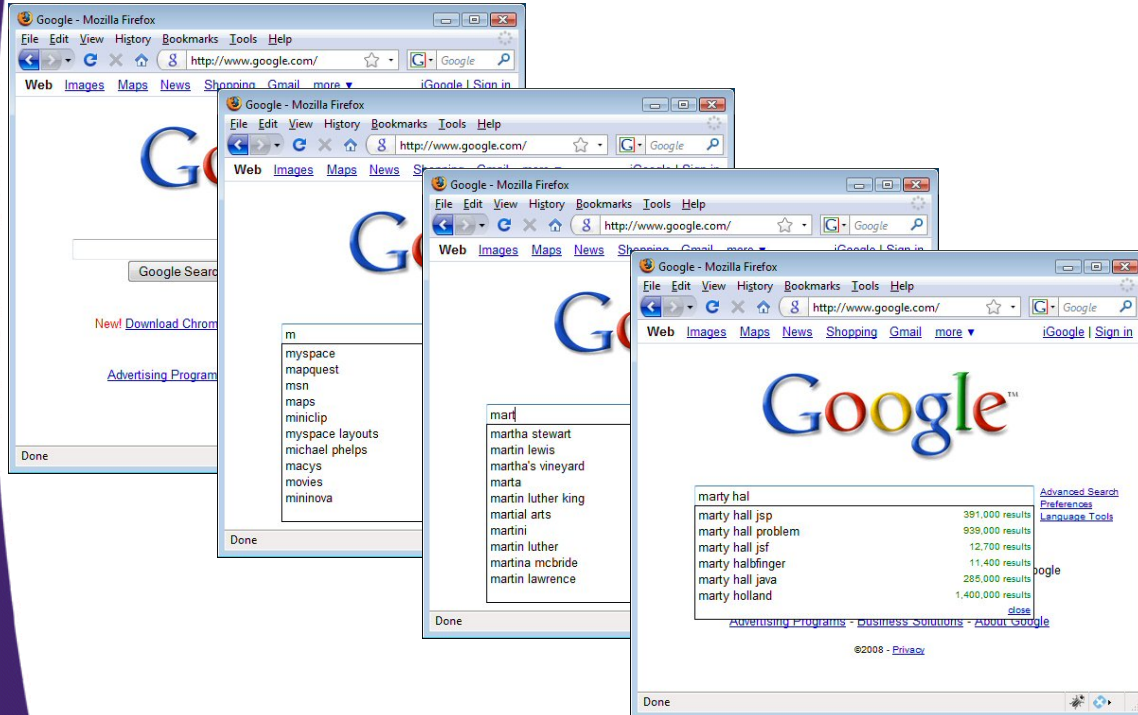
Frequent small updates



8

Diagrams from *Ajax in Action* by Dave Crane et al. Manning Press. See recommended books in next lecture.

# Google Home Page (formerly Google Suggest)



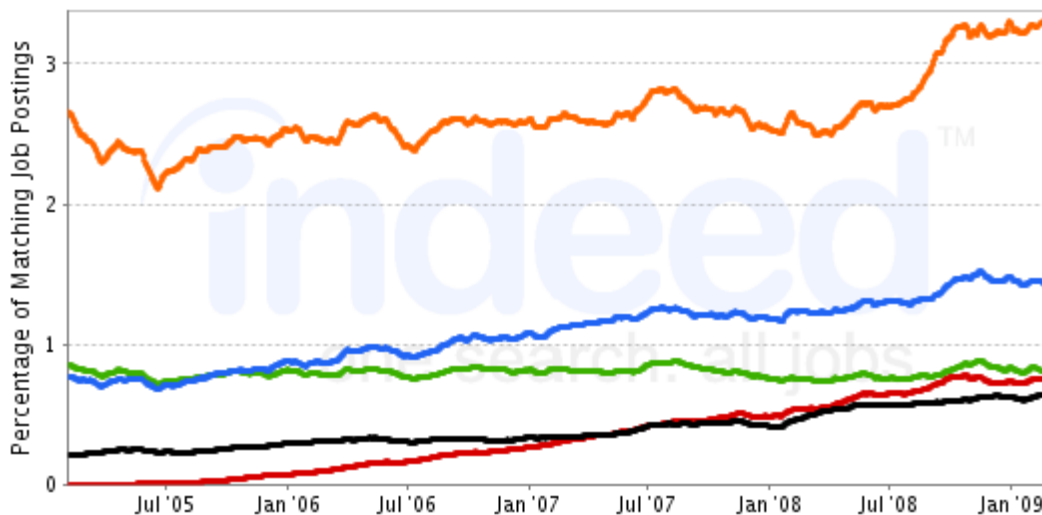
9

# Ajax Jobs (as of Jan. 2009)

- **From indeed.com**
  - Claims to compile data from most major job sites

Job Trends from Indeed.com

— java — c# — vb — php — ajax



10

# Why Ajax4jsf?

- **There are many Ajax libraries**
  - Why do we need another one?
- **Niche for Ajax4jsf**
  - For applications that are doing hybrid apps
    - Sometimes submitting forms and doing normal page navigation
    - Sometimes using Ajax to update parts of page
  - For apps that want integration with JSF
    - Ajax parts know about JSF managed beans
    - Can use JSF tags and tools with Ajax
    - Integration of approaches for normal Web apps and Ajax
  - Also filled by other JSF-Ajax libraries
    - **JSF 2.0!** (See tutorial at [coreservlets.com](http://coreservlets.com))
    - Other parts of RichFaces
    - IceFaces
    - Parts of Oracle ADF
    - Parts of Apache Tomahawk
    - Backbase Framework for JSF

11

© 2012 Marty Hall



## Installation and Setup

**Customized Java EE Training: <http://courses.coreservlets.com/>**

Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Hadoop, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

# Installing Ajax4jsf

- **Download latest binary version**
  - <http://labs.jboss.com/jbossajax4jsf/downloads/>
- **Unzip**
  - Into any location
- **Install ajax4jsf.jar**
  - Copy /lib/ajax4jsf.jar into the WEB-INF/lib directory of your Web app
- **Add filter to web.xml**
  - See next page
- **Shortcut**
  - Download ajax4jsf-blank from <http://www.coreservlets.com/JSF-Tutorial/>
    - Web app with all MyFaces and Ajax4jsf JAR files and web.xml entries

13

# Filter Settings for web.xml

```
<?xml version="1.0"?>
<web-app ...>
  ...
  <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>
</web-app>
```

14

# Using Ajax4jsf

- **Use xhtml**
  - Most Ajax applications use xhtml, not HTML 4
- **Add taglib for ajax4jsf**
  - `<%@ taglib uri="https://ajax4jsf.dev.java.net/ajax" prefix="a4j"%>`
- **Give ids to sections that you want to update**
  - `<h:outputText ... id="someName"/>`
    - Note: ids within a page must be unique
- **Use a4j: tags**
  - Almost all *must* go inside h:form
    - Even the ones that don't use form elements (e.g., a4j:poll)

15

# Basic Template

```
<?xml version="1.0" encoding="UTF-8"?>
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="https://ajax4jsf.dev.java.net/ajax"
    prefix="a4j"%>
<f:view>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<meta http-equiv="content-type"
    content="text/html; charset=UTF-8" />
<head><title>Some Title</title>
</head>
<body>
...
</body></html>
</f:view>
```

XML header recommended by xhtml specification, but you should omit because of Internet Explorer bug.

16

## XHTML: Case

- **In HTML 4, case does not matter for tag names and attribute names**
  - `<BODY>`, `<Body>`, and `<body>` are equivalent
  - `<H1 ALIGN="...">` is equivalent to `<H1 aLiGn="...">`
- **In xhtml, tag names and attribute names must be in lower case**
  - `<body>`, `<h1 align="...">`

17

## XHTML: Quotes

- **In HTML 4, quotes are optional if attribute value contains only alphanumeric values**
  - `<H1 ALIGN="LEFT">` or `<H1 ALIGN=LEFT>`
- **In xhtml, you must always use single or double quotes**
  - `<h1 align="left">` or `<h1 align='left'>`

18

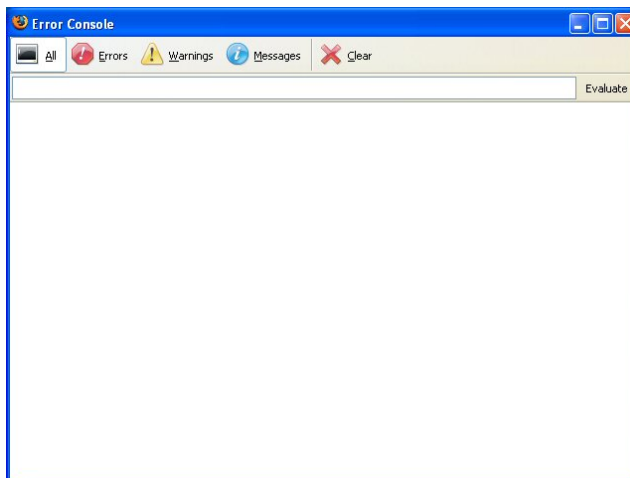
# XHTML: End Tags

- **HTML 4**
  - Some tags are containers
    - `<H1>...</H1>`, `<A HREF...>...</A>`
  - Some tags are standalone
    - `<BR>`, `<HR>`
  - Some tags have optional end tags
    - `<P>`, `<LI>`, `<TR>`, `<TD>`, `<TH>`
- **XHTML**
  - All tags are containers. End tags always required.
    - `<p>...</p>`, `<li>...</li>`
  - If there is no body content, start/end tags can be merged
    - `<br></br>` or just `<br/>`

19

# Ajax and the Firefox JavaScript Console

- **Invoke with Control-Shift-J**



- **Also see Venkman JavaScript debugger**
  - <http://www.mozilla.org/projects/venkman/>
  - <https://addons.mozilla.org/firefox/216/>

20



# Main Ajax4jsf Tags

Customized Java EE Training: <http://courses.coreservlets.com/>

Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Hadoop, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

## Tag Summary

- **a4j:commandButton** and **a4j:commandLink**
  - Run code on the server, then update specified JSF element (or comma separated element list) after.
    - `<a4j:commandButton action="#{bean.method}" value="Button label" reRender="some-id"/>`  
 ... `<h:outputText value="#{bean.prop}" id="some-id"/>`
- **a4j:poll**
  - Run code periodically on server, then update specified JSF element(s)
- **a4j:support**
  - Capture JavaScript event in *any* existing JSF control and invoke server-side code, then update specified element(s)
    - `<h:inputText ...>`  
`<a4j:support event="onkeyup" reRender="some-id"/>`  
`</h:inputText>`

## a4j:commandButton: Basic Syntax

```
<a4j:commandButton  
  action="#{bean.method}"  
  value="Button label"  
  reRender="some-id"/>
```

When you press the button, send a behind-the-scenes, asynchronous HTTP call to server, and run this method. This method should look like a normal action controller (i.e., no arguments and returns a String), but the return value is ignored (i.e., is not used to match a navigation rule in faces config).

After action method is executed, re-evaluate and re-display the JSF element that has this id. The point is that this JSF element should output something that changed as a result of the call to the action controller.

23

## Reminder: Template

```
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f" %>  
<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h" %>  
<%@ taglib uri="https://ajax4jsf.dev.java.net/ajax" prefix="a4j"%>  
<f:view>  
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"  
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  
<html xmlns="http://www.w3.org/1999/xhtml">  
<head>  
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />  
<link rel="stylesheet"  
  href="/css/styles.css"  
  type="text/css"/>  
<title>Ajax4JSF Examples</title>  
</head>  
<body> ...
```

24

## h:commandButton: Example

```
<tr><td width="100">
  <h:form>
    <a4j:commandButton
      action="#{numBean.makeResult}"
      value="Show Random Number"
      reRender="region1"/>
  </h:form>
</td>
<td width="100">
  <h:outputText value="#{numBean.result}"
    id="region1"/>
</td>
</tr>
```

25

## h:commandLink: Example

```
<tr><td width="100">
  <h:form>
    <a4j:commandLink
      action="#{numBean.makeResult}"
      value="Show Random Number"
      reRender="region2"/>
  </h:form>
</td>
<td width="100">
  <h:outputText value="#{numBean.result}"
    id="region2"/>
</td>
</tr>
```

Remember that ids must be unique within a page.

26

## a4j:commandButton and a4j:commandLink: Bean

```
public class RandomNumberBean {
    private int range = 1;
    private double result;

    public int getRange() {
        return(range);
    }

    public void setRange(int range) {
        this.range = range;
    }

    public String makeResult() {
        result = Math.random() * range;
        return(null);
    }

    public double getResult() {
        return(result);
    }
}
```

Action controllers must have String return type to satisfy JSF API.

However, return values are ignored, since navigation rules do not apply.

27

## a4j:commandButton and a4j:commandLink: faces-config

```
<managed-bean>
  <managed-bean-name>
    numBean
  </managed-bean-name>
  <managed-bean-class>
    coreservlets.RandomNumberBean
  </managed-bean-class>
  <managed-bean-scope>
    request
  </managed-bean-scope>
</managed-bean>
```

28

## a4j:commandButton: Example 2

- **a4j:commandButton** results in normal JSF request processing cycle
  - So, form is still submitted
  - Bean setter methods still execute
  - Action controller runs *after* setter methods, so controller has access to results of all form fields

29

## a4j:commandButton: Example 2

```
<tr><td width="100">
  <h:form>
    Range:
    <h:inputText value="#{numBean.range}"
      size="5"/><br/>
    <a4j:commandButton
      action="#{numBean.makeResult}"
      value="Show Random Number"
      reRender="region3"/>
  </h:form>
</td>
<td width="100">
  <h:outputText value="#{numBean.result}"
    id="region3"/>
</td>
</tr>
```

30

# a4j:commandButton and a4j:commandLink: Bean

```
public class RandomNumberBean {
    private int range = 1;
    private double result;

    public int getRange() {
        return(range);
    }

    public void setRange(int range) {
        this.range = range;
    }

    public String makeResult() {
        result = Math.random() * range;
        return(null);
    }

    public double getResult() {
        return(result);
    }
}
```

31

# a4j:commandButton and a4j:commandLink: Results

The screenshot shows two overlapping browser windows displaying the results of a4j:commandButton and a4j:commandLink actions. The top window shows the initial state with a range of 1 and a result of 0.0. The bottom window shows the results after clicking the buttons, with a range of 150 and a result of 129.31059414654032.

Control	Result
Show Random Number	0.0
Show Random Number	0.0
Range: 1	0.0
Show Random Number	0.0

Control	Result
Show Random Number	0.8080057864819878
Show Random Number	0.5656789493724107
Range: 150	129.31059414654032
Show Random Number	

32

## Limitations on Use of h:outputText with Ajax4jsf

- **In JSF, the following is perfectly legal**
  - `<body bgcolor=  
"<h:outputText #{myBean.fgColor}"/>">`
  - Results in `<body bgcolor="red">`
- **But the following is illegal**
  - `<body bgcolor=  
"<h:outputText #{myBean.fgColor} id="foo"/>">`
  - Results in `<body bgcolor="<span id="foo">red</span>">`
- **In regular JSF, this is no problem**
  - Since there is no need for ids
- **But with Ajax, you need a DOM-accessible region to update**
  - So, you are limited in the places that you can update asynchronously
  - You can only update separate xhtml elements, not just sections

33

## a4j:poll: Basic Syntax

```
<a4j:poll  
  interval="x"  
  reRender="some-id"/>
```


- **Interpretation**
  - Every *x* milliseconds, send off an asynchronous HTTP request to the server.
  - Re-evaluate and re-display the JSF element with the id *some-id*.
    - Key point: this element should result from code that gives different values at different times
    - Unlike with `a4j:commandButton`, there is no explicit additional server-side method to run

34

## a4j:poll: Example

```
<h:form>
  <a4j:poll interval="5000"
            reRender="timeDisplay"/>
  <h2>
    <h:outputText value="#{timeBean.time}"
                  id="timeDisplay"/>
  </h2>
</h:form>
```

The getTime method  
returns different results at  
different times.



35

## a4j:poll: Bean

```
public class TimeBean {
    public Date getTime() {
        return(new Date());
    }
}
```

36

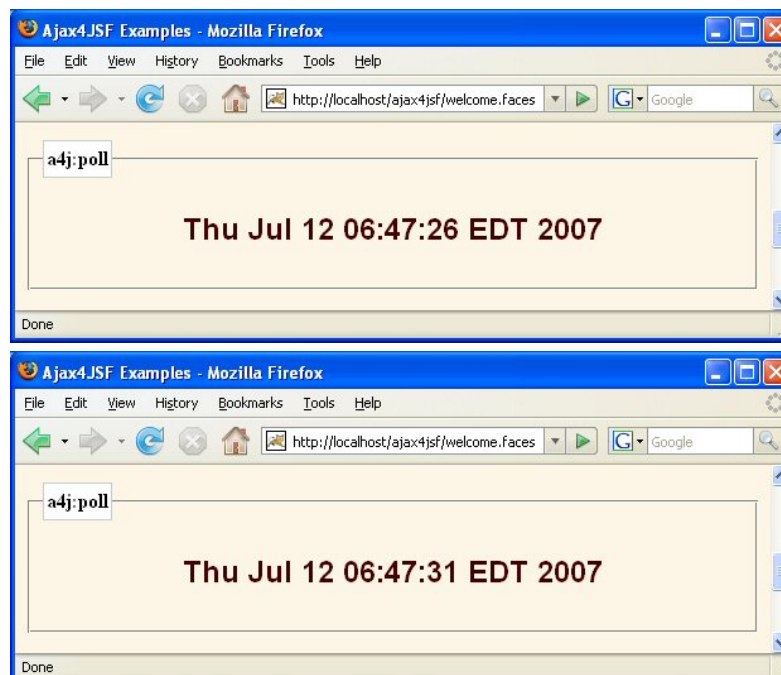
# a4j:poll: faces-config.xml

```
<managed-bean>
  <managed-bean-name>
    timeBean
  </managed-bean-name>
  <managed-bean-class>
    coreservlets.TimeBean
  </managed-bean-class>
  <managed-bean-scope>
    application ←
  </managed-bean-scope>
</managed-bean>
```

TimeBean has no state, so for efficiency, reuse the same instance.

37

# a4j:poll: Results



38

## a4j:support: Basic Syntax

```
<h:someTag ...>  
  <a4j:support event="javascript-event"  
              reRender="some-id">  
</h:someTag>
```

- **Interpretation**

- Do whatever someTag normally does, but if the specified JavaScript event (onclick, onchange, onkeypress, etc.) occurs, send a behind-the-scenes asynchronous HTTP request to the server that results in the specified JSF element being re-evaluated and re-displayed
- Note: form values are sent and setter methods are run

39

## JavaScript Event Handlers used with a4j:support

- **onchange**
  - User changes element (IE: and element loses focus)
- **onclick/onclick**
  - User single/double clicks form element or link
- **onfocus/onblur**
  - Element receives/loses focus
- **onkeydown/onkeypress/onkeyup**
  - User presses/presses-or-holds/releases a key
- **onmousedown/onmouseup**
  - User presses/releases mouse
- **onmousemove**
  - User moves mouse
- **onmouseover/onmouseout**
  - User moves mouse onto/off area or link
- **onselect**
  - User selects text within a textfield or textarea
- **onsubmit**
  - User submits form

40

# a4j:support: Example 1

- **Idea**
  - Use h:inputText to make a textfield
  - As the user types into the textfield, copy the value into regular text
- **Approach**
  - Textfield
    - Stores result in myBean.message
    - `<a4j:support event="onkeyup" reRender="output-region"/>`
  - Separate output field
    - `<h:outputText value="#{myBean.message}" id="output-region"/>`
- **Note**
  - Bundled with ajax4jsf-blank
    - <http://www.coreservlets.com/JSF-Tutorial/>

41

# a4j:support: Example

```
<h:form>
  <table border="1">
    <tr><th>Textfield</th>
      <th>Ajax Value</th>
    </tr>
    <tr><td width="100">
      <h:inputText value="#{myBean.message}">
        <a4j:support event="onkeyup"
          reRender="output-region"/>
      </h:inputText></td>
      <td width="100">
        <h:outputText value="#{myBean.message}"
          id="output-region"/></td>
      </tr>
    </table>
  </h:form>
```

42

## a4j:support: Bean

```
public class MessageBean {
    private String message;

    public String getMessage() {
        return(message);
    }

    public void setMessage(String message) {
        this.message = message;
    }
}
```

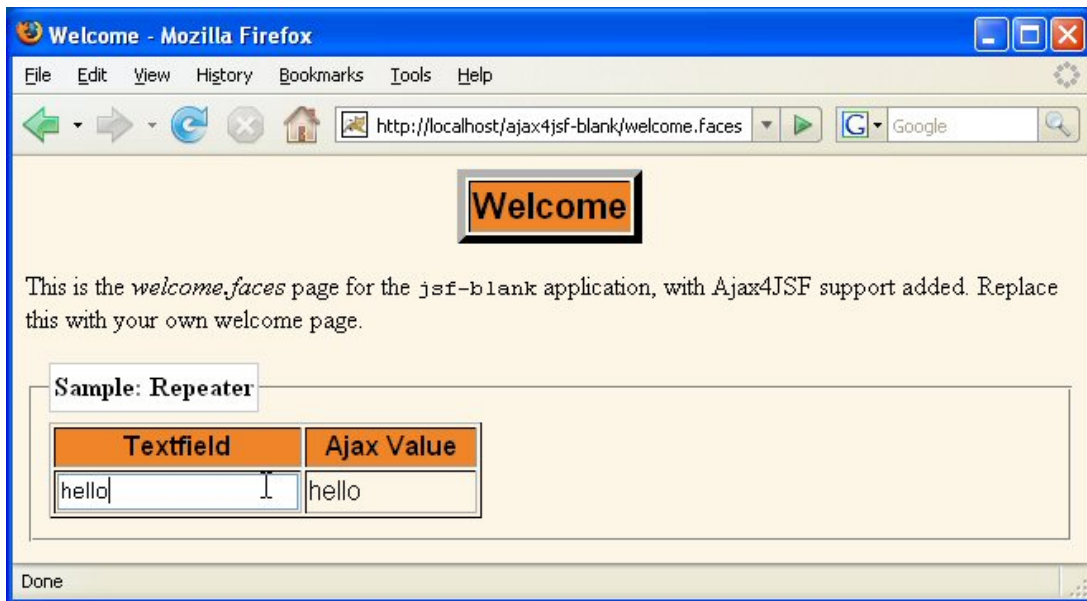
43

## a4j:support: faces-config.xml

```
<managed-bean>
    <managed-bean-name>
        myBean
    </managed-bean-name>
    <managed-bean-class>
        coreservlets.MessageBean
    </managed-bean-class>
    <managed-bean-scope>
        request
    </managed-bean-scope>
</managed-bean>
```

44

# a4j:support: Results



45

# a4j:support: Example 2

- **Idea**
  - Use `h:selectOneMenu` to make a list of US states
  - When the user selects a state, a list of corresponding cities is shown (again, using `h:selectOneMenu`)
  - When city selected, population of that city is displayed
- **Approach**
  - State list
    - `<a4j:support event="onchange" reRender="city-list"/>`
  - City List
    - `<a4j:support event="onchange" reRender="pop-field"/>`
  - Bean
    - **Make it session scoped** so values persist across across multiple submissions (since there are at least two)

46

## a4j:support: Example

```
<h:form>
  State:
  <h:selectOneMenu value="#{locationBean.state}">
    <f:selectItems value="#{locationBean.states}"/>
    <a4j:support event="onchange" reRender="cityList"/>
  </h:selectOneMenu><br/>
  City:
  <h:selectOneMenu value="#{locationBean.city}"
    disabled="#{locationBean.cityListDisabled}"
    id="cityList">
    <f:selectItems value="#{locationBean.cities}"/>
    <a4j:support event="onchange" reRender="population"/>
  </h:selectOneMenu><br/>
  Population:
  <h:outputText value="#{locationBean.city}"
    escape="false"
    id="population"/></h:form>
```

47

## a4j:support: Bean

```
public class LocationBean implements Serializable {
  private String state;
  private String city;
  private boolean isCityListDisabled = true;

  public String getState() { return (state); }

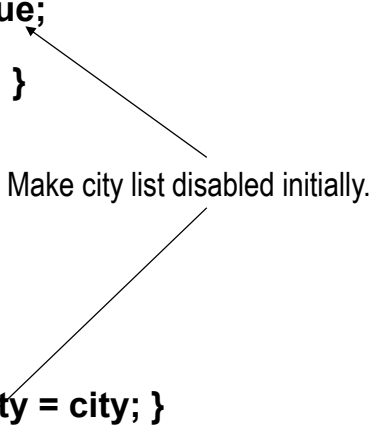
  public void setState(String state) {
    this.state = state;
    isCityListDisabled = false;
  }

  public String getCity() { return(city); }

  public void setCity(String city) { this.city = city; }

  public boolean isCityListDisabled() {
    return(isCityListDisabled);
  }
}
```

Make city list disabled initially.



48

## a4j:support: Bean (Continued)

```
...
public List<SelectItem> getStates() {
    List<SelectItem> states =
        new ArrayList<SelectItem>();
    states.add(
        new SelectItem("--- Select State ---"));
    for(StateInfo stateData:
        StateInfo.getNearbyStates()) {
        states.add(
            new SelectItem(stateData.getStateName()));
    }
    return(states);
}
```

Put dummy value at the top of the list so that any real user selection is considered a change

49

## a4j:support: Bean (Continued)

```
public SelectItem[] getCities() {
    SelectItem[] cities =
        { new SelectItem("--- Choose City ---")};
    if(!isCityListDisabled && (state != null)) {
        for(StateInfo stateData:
            StateInfo.getNearbyStates()) {
            if(state.equals(stateData.getStateName())) {
                cities = stateData.getCities();
                break;
            }
        }
    }
    return(cities);
}
```

Result of form submission  
(i.e., value from the state list)

Value is a SelectItem array

50

## a4j:support: Supporting Class (StateInfo)

```
public class StateInfo {
    private String stateName;
    private SelectItem[] cities;

    public StateInfo(String stateName,
                     SelectItem...cities) {
        this.stateName = stateName;
        this.cities = cities;
    }

    public String getStateName() {
        return(stateName);
    }

    public SelectItem[] getCities() {
        return(cities);
    }
}
```

Value is a SelectItem array

51

## a4j:support: Supporting Class (StateInfo) Continued

```
private static StateInfo[] nearbyStates =
    { new StateInfo("Maryland",
        new SelectItem("<i>unknown</i>",
            "--- Choose City ---"),
        new SelectItem("635815", "Baltimore"),
        new SelectItem("57907", "Frederick"),
        new SelectItem("57698", "Gaithersburg"),
        new SelectItem("57402", "Rockville")),
    ...
}
```

Values that are used when an entry is selected and form is submitted. These are the values displayed in the text when the city field changes

Values that are displayed in the list

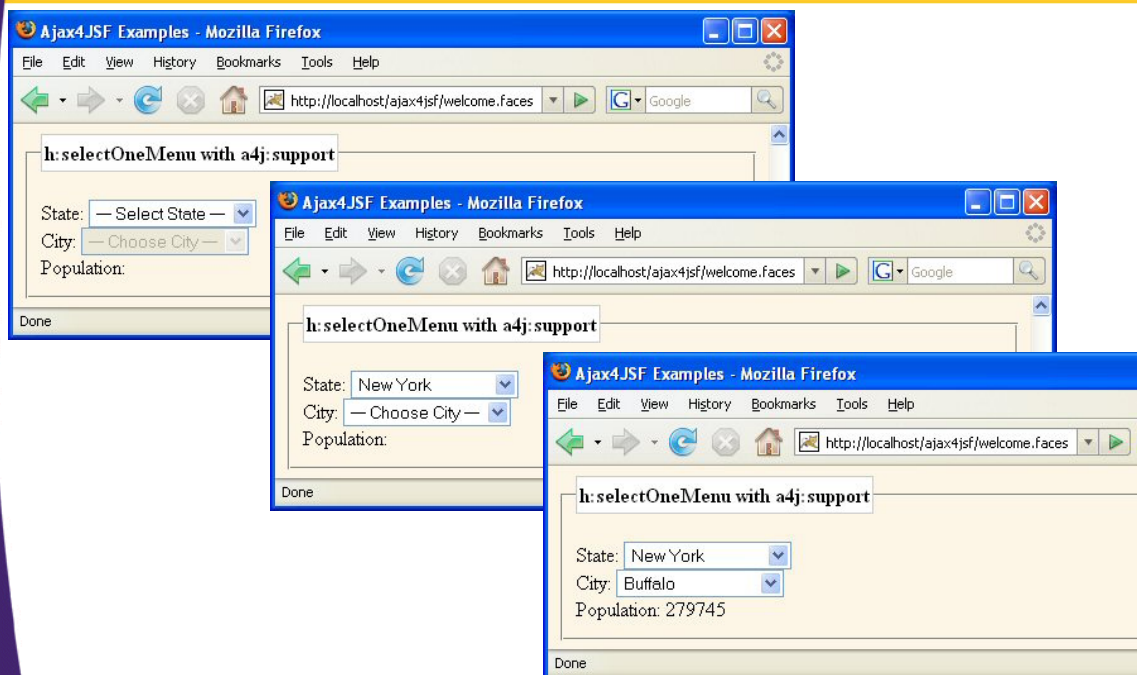
52

# a4j:support: faces-config.xml

```
<managed-bean>
  <managed-bean-name>
    locationBean
  </managed-bean-name>
  <managed-bean-class>
    coreservlets.LocationBean
  </managed-bean-class>
  <managed-bean-scope>
    session
  </managed-bean-scope>
</managed-bean>
```

53

# a4j:support: Results



54



## Other Ajax Tools

**Customized Java EE Training:** <http://courses.coreservlets.com/>

Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Hadoop, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

## Client-Side Tools (JavaScript Libraries with Ajax Support)

- **Prototype**
  - <http://www.prototypejs.org/>
- **Dojo**
  - <http://www.dojotoolkit.org/>
- **script.aculo.us**
  - <http://script.aculo.us/>
- **ExtJS**
  - <http://extjs.com/>
- **Yahoo User Interface Library (YUI)**
  - <http://developer.yahoo.com/yui/>

## Server-Side Tools

- **Direct Web Remoting**
  - Lets you call Java methods semi-directly from JavaScript
  - <http://getahead.ltd.uk/dwr/>
- **JSON/JSON-RPC**
  - For sending data to/from JavaScript with less parsing
  - <http://www.json.org/>
  - <http://json-rpc.org/>

57

## Hybrid Client/Server Tools

- **JSP custom tag libraries**
  - Create tags that generate HTML and JavaScript
  - <http://courses.coreservlets.com/Course-Materials/msajsp.html>
- **AjaxTags (built on top of [script.aculo.us](http://script.aculo.us))**
  - JSP custom tags that generate Ajax functionality
    - Supports many powerful Ajax capabilities with very simple syntax
    - <http://ajaxtags.sourceforge.net>
- **Google Web Toolkit**
  - Write code in Java, translate it to JavaScript
    - <http://code.google.com/webtoolkit/>
  - Also see <https://ajax4jsf.dev.java.net/>
    - GWT/JSF Integration Toolkit

58

## JSF-Based Tools

- **Trinidad (formerly Oracle ADF)**
  - <http://www.oracle.com/technology/products/jdev/htdocs/partners/addins/exchange/jsf/> (also [myfaces.apache.org](http://myfaces.apache.org))
- **Tomahawk**
  - <http://myfaces.apache.org/tomahawk/>
- **Ajax4JSF (and RichFaces)**
  - <http://labs.jboss.com/jbossajax4jsf/>
- **IceFaces**
  - <http://www.icefaces.org/>
- **Build your own**
  - <http://courses.coreservlets.com/Course-Materials/jsf.html>

59

## Tools and Toolkits

- **Client-Side Tools (JavaScript Libraries)**
  - Dojo
    - <http://www.dojotoolkit.org/>
  - Google Web Toolkit
    - Write code in Java, translate it to JavaScript
      - <http://code.google.com/webtoolkit/>
    - Also see G4JSF
      - GWT/JSF Integration Toolkit
      - <https://ajax4jsf.dev.java.net/nonav/ajax/gwt/gwt-cdk.html>
  - [script.aculo.us](http://script.aculo.us/)
    - <http://script.aculo.us/>
  - ExtJS
    - <http://extjs.com/>
  - Yahoo User Interface Library (YUI)
    - <http://developer.yahoo.com/yui/>

60

# Summary

- **a4j:commandButton/a4j:commandLink**
  - `<a4j:commandButton action="#{bean.method}" value="Button label" reRender="some-id"/>`
  - `<a4j:commandLink action="#{bean.method}" value="Link text" reRender="some-id"/>`
- **a4j:poll**
  - `<a4j:poll interval="x-milliseconds" reRender="some-id"/>`
- **a4j:support**
  - `<h:someTag ...>`
    - `<a4j:support event="javascript-event" reRender="some-id">`
  - `</h:someTag>`

61

© 2012 Marty Hall



## Questions?

Customized Java EE Training: <http://courses.coreservlets.com/>

Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Hadoop, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.