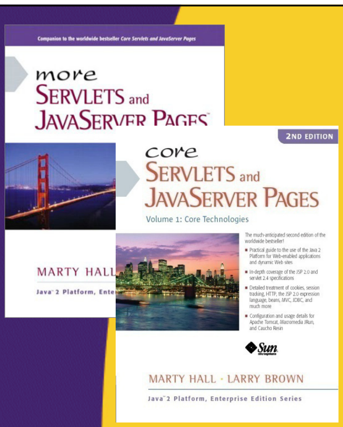
The Web Application Life-Cycle Events Framework

Originals of Slides and Source Code for Examples:
http://courses.coreservlets.com/Course-Materials/msajsp.html



Agenda

- **Reason for listeners**
- **Monitoring creation and destruction of the servlet context**
- **Detecting changes in servlet context attributes**
- **Recognizing session creation and destruction**
- **Watching for changes in session attributes**
- **Combining activities**

Life-Cycle Events Framework

- **Motivation 1: Respond to events in the overall application life cycle.**
  – Notice when certain attributes are removed from the servlet context.
- **Motivation 2: Perform tasks more general than any one servlet or JSP page has responsibility for.**
  – Insert application-wide data into the servlet context before any servlets or JSP pages are accessed
  – Keep track of certain session attributes, regardless of which servlet or JSP page added the attributes
- **New with servlets 2.3.**
  – Note: this section has not been updated to 2.4 or 2.5!

Available Listeners

- **Servlet context listeners.**
  - These listeners are notified when the servlet context (i.e., the Web application) is initialized and destroyed.
- **Servlet context attribute listeners.**
  - These listeners are notified when attributes are added to, removed from, or replaced in the servlet context.
- **Session listeners.**
  - These listeners are notified when session objects are created, invalidated, or timed out.
- **Session attribute listeners.**
  - These listeners are notified when attributes are added to, removed from, or replaced in any session.

General Implementation Strategy

1. **Implement the appropriate interface.**
   - Use ServletContextListener, ServletContextAttributeListener, HttpSessionListener, or HttpSessionAttributeListener.
2. **Override the methods needed to respond to the events of interest.**
   - Provide empty bodies for the other methods in the interface.
3. **Access the important Web application objects.**
   - Six objects that you are likely to use in event-handling methods:
     - The servlet context
     - The name of the servlet context attribute that changed
     - The value of the servlet context attribute that changed
     - The session object
     - The name of the session attribute that changed

General Implementation Strategy (Continued)

4. **Use these objects.**
   – This process is application specific, but there are some common themes. For example, with the servlet context, you are most likely to read initialization parameters (getInitParameter), store data for later access (setAttribute), and read previously stored data (getAttribute).
5. **Declare the listener.**
   – You do this with the `listener` and `listener-class` elements of the general Web application deployment descriptor (*web.xml*) or of a tag library descriptor file.
6. **Provide any needed initialization parameters.**
   – Servlet context listeners commonly read context initialization parameters to use as the basis of data that is made available to all servlets and JSP pages. You use the `context-param` *web.xml* element to provide the names and values of these initialization parameters.

Monitoring Creation and Destruction of the Servlet Context

- **The ServletContextListener class responds to the initialization and destruction of the servlet context.**
  – These events correspond to the creation and shutdown of the Web application itself.
- **ServletContextListener is most commonly used to**
  – Set up application-wide resources like database connection pools
  – Read the initial values of application-wide data that will be used by multiple servlets and JSP pages.

Specific Implementation Strategy: Servlet Context Listeners

1. **Implement the ServletContextListener interface.**
2. **Override contextInitialized and contextDestroyed.**
   - **contextInitialized** is triggered when the Web application is first loaded and the servlet context is created. Most common tasks:
     - Creating application-wide data (e.g., by reading context init params)
     - Storing that data in an easily accessible location .
   - **contextDestroyed** is triggered when the Web application is being shut down and the servlet context is about to be destroyed. Most common task:
     - Releasing resources (e.g. closing connections).
3. **Obtain a reference to the servlet context.**
   - The contextInitialized and contextDestroyed methods each take a ServletContextEvent as an argument.
   - The ServletContextEvent class has a getServletContext method

Specific Implementation Strategy: Servlet Context Listeners (Continued)

4. **Use the servlet context.**
   - Read initialization parameters: getInitParameter
   - Store data:setAttribute
   - Make log file entries: log.
5. **Declare the listener.**

```
<listener>
  <listener-class>package.Listener</listener-class>
</listener>
```

6. **Provide needed initialization parameters.**

```
<context-param>
  <param-name>name</param-name>
  <param-value>value</param-value>
</context-param>
```

Example: Web Site for Company with Changing Name

```
public class InitialCompanyNameListener
    implements ServletContextListener {
  private static final String DEFAULT_NAME =
    "MISSING-COMPANY-NAME";

 public void contextInitialized(ServletContextEvent event) {
    ServletContext context = event.getServletContext();
    setInitialAttribute(context,
                        "companyName",
                        DEFAULT_NAME);
    setInitialAttribute(context,
                        "formerCompanyName",
                        "");
  }

  public void contextDestroyed(ServletContextEvent event) {}
```

Servlet Context Listener Example (Continued)

```
    private void setInitialAttribute(ServletContext context,
                                     String initParamName,
                                     String defaultValue) {
      String initialValue =
        context.getInitParameter(initParamName);
      if (initialValue != null) {
        context.setAttribute(initParamName, initialValue);
      } else {
        context.setAttribute(initParamName, defaultValue);
      }
    }
  …
  }
```

Servlet Context Listener: web.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE …>

<web-app>
  <context-param>
    <param-name>companyName</param-name>
    <param-value>not-dot-com.com</param-value>
  </context-param>

  <context-param>
    <param-name>formerCompanyName</param-name>
    <param-value>hot-dot-com.com</param-value>
  </context-param>
  …
```

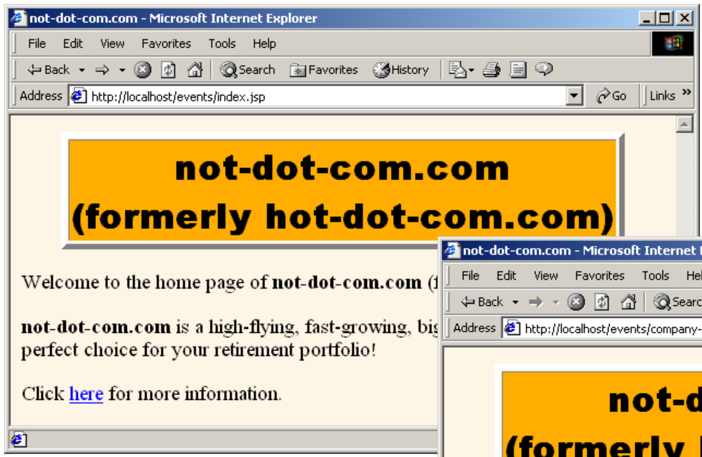
Servlet Context Listener: web.xml (Continued)

```
  …

  <listener>
    <listener-class>
      moreservlets.listeners.InitialCompanyNameListener
    </listener-class>
  </listener>

</web-app>
```

Servlet Context Listener: Using Results in JSP Technology

```
<%
String companyName =
  InitialCompanyNameListener.getCompanyName(application);
String formerCompanyName =
  InitialCompanyNameListener.getFormerCompanyName(application);
… %>
<TITLE><%= companyName %></TITLE>
…
<TABLE BORDER=5 ALIGN="CENTER">
  <TR><TH CLASS="TITLE">
       <%= companyName %><BR>
       <%= formerCompanyDescription %>
…
<UL>
  <LI><A HREF="products.jsp"><%= companyName %> products</A>
  <LI><A HREF="services.jsp"><%= companyName %> services</A>
  <LI><A HREF="history.jsp"><%= companyName %> history</A>
  <LI><A HREF="invest.jsp">investing in <%= companyName %></A>
  <LI><A HREF="contact.jsp">contacting <%= companyName %></A>
</UL>
…
```

Servlet Context Listener: Results


not-dot-com.com - Microsoft Internet Explorer

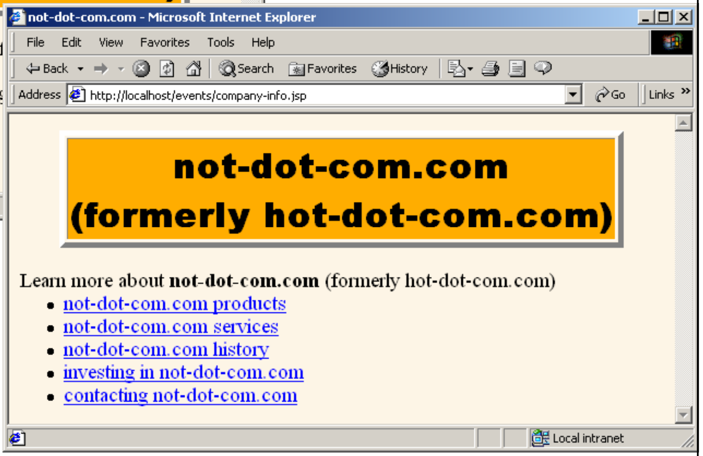
Address http://localhost/events/index.jsp

not-dot-com.com
(formerly hot-dot-com.com)

Welcome to the home page of **not-dot-com.com** (f

not-dot-com.com is a high-flying, fast-growing, big
perfect choice for your retirement portfolio!

Click here for more information.


not-dot-com.com - Microsoft Internet Explorer

Address http://localhost/events/company-info.jsp

not-dot-com.com
(formerly hot-dot-com.com)

Learn more about **not-dot-com.com** (formerly hot-dot-com.com)

- not-dot-com.com products
- not-dot-com.com services
- not-dot-com.com history
- investing in not-dot-com.com
- contacting not-dot-com.com

Detecting Changes in Servlet Context Attributes

- **ServletContextListener**
  - Lets you set up *initial* values of resources and store references to them in the servlet context.
- **But what if you want to be notified whenever these resources change?**
  - For example, what if the value of resource B depends on the value of resource A? If resource A changes, you need to automatically update the value of resource B.
  - Handling this situation is the job of servlet context *attribute* listeners.

Implementation Strategy: Servlet Context Attribute Listeners

1. **Implement ServletContextAttributeListener**
2. **Override attributeAdded, attributeReplaced, and attributeRemoved.**
   - attributeAdded is triggered when a new attribute name is first added to the servlet context.
   - attributeReplaced is triggered when a new value is assigned to an existing name. attributeAdded is *not* triggered in this case. The old value is obtained via event.getValue and the new value is obtained via context.getAttribute.
   - attributeRemoved is triggered when a servlet context attribute is removed altogether.
3. **Obtain references to the attribute name, attribute value, and servlet context.**
   - Call the following methods of the event object: getName, getValue, and getServletContext

Implementation Strategy: Servlet Context Attribute Listeners (Continued)

4. **Use the objects.**
   – You normally compare attribute name to a stored name to see if it is the one you are monitoring. The attribute value is used in an application-specific manner. The servlet context is usually used to read previously stored attributes (getAttribute), store new or changed attributes (setAttribute), and make entries in the log file (log).
5. **Declare the listener.**
   – Use the `listener` and `listener-class` elements to list the fully qualified name of the listener class,

```
<listener>
  <listener-class>
    somePackage.SomeListener
  </listener-class>
</listener>
```

Example: Updating Former Company Name

```
public class ChangedCompanyNameListener
    implements ServletContextAttributeListener {

  public void attributeAdded
         (ServletContextAttributeEvent event) {}

  public void attributeRemoved
         (ServletContextAttributeEvent event) {}
```

Example: Updating Former Company Name (Continued)

```
public void attributeReplaced
           (ServletContextAttributeEvent event) {
  if (event.getName().equals("companyName")) {
    String oldName = (String)event.getValue();
    ServletContext context =
      event.getServletContext();
    context.setAttribute("formerCompanyName",
                         oldName);
  }
}
```

Example: Updating Former Company Name (Continued)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE …>

<web-app>
  <listener>
    <listener-class>
    moreservlets.listeners.ChangedCompanyNameListener
    </listener-class>
  </listener>
```

Example: Updating Former Company Name (Continued)

```
<servlet>
  <servlet-name>
    ChangeCompanyName
  </servlet-name>
  <servlet-class>
    moreservlets.ChangeCompanyName
  </servlet-class>
</servlet>

<servlet>
  <servlet-name>Redirector</servlet-name>
  <servlet-class>
    moreservlets.RedirectorServlet
  </servlet-class>
</servlet>
```

Example: Updating Former Company Name (Continued)

```
<servlet-mapping>
  <servlet-name>
    ChangeCompanyName
  </servlet-name>
  <url-pattern>
    /admin/ChangeCompanyName
  </url-pattern>
</servlet-mapping>

<servlet-mapping>
  <servlet-name>Redirector</servlet-name>
  <url-pattern>/servlet/*</url-pattern>
</servlet-mapping>
```

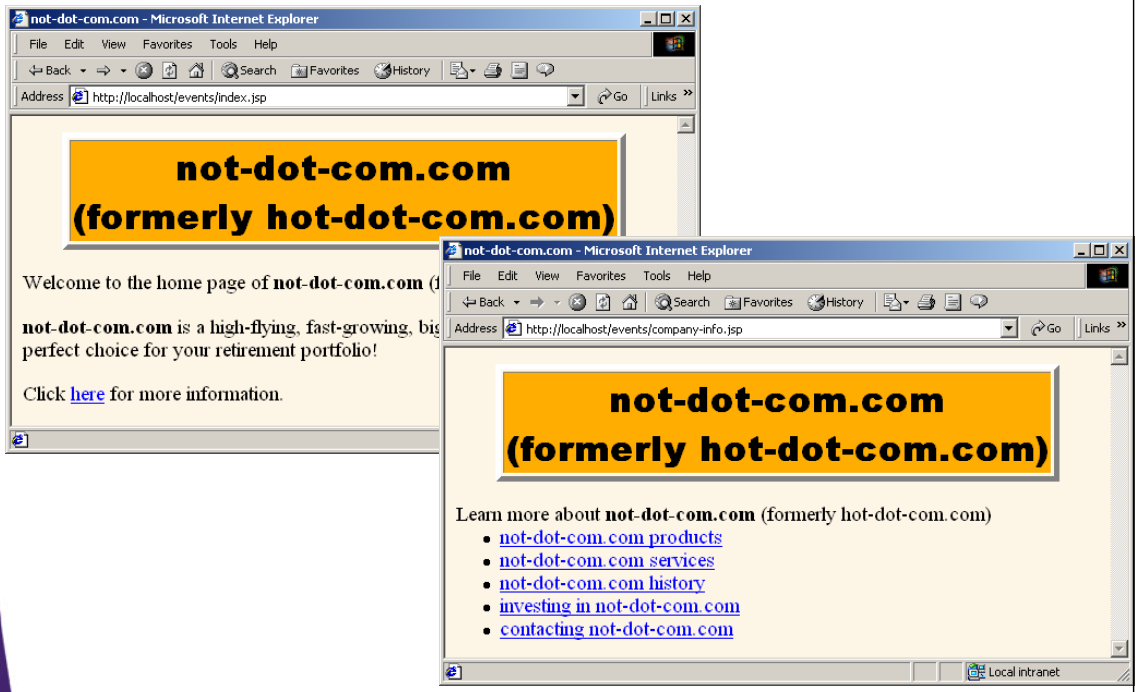
Example: Updating Former Company Name (Continued)

```
<security-constraint>
  <web-resource-collection>
    <web-resource-name>Admin</web-resource-name>
    <url-pattern>/admin/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <role-name>ceo</role-name>
  </auth-constraint>
</security-constraint>
```

Example: Updating Former Company Name (Continued)

```
<login-config>
  <auth-method>FORM</auth-method>
  <form-login-config>
    <form-login-page>
      /admin/login.jsp
    </form-login-page>
    <form-error-page>
      /admin/login-error.jsp
    </form-error-page>
  </form-login-config>
</login-config>
```

Example: Updating Former Company Name (Results)



not-dot-com.com - Microsoft Internet Explorer

http://localhost/events/index.jsp

not-dot-com.com (formerly hot-dot-com.com)

Welcome to the home page of **not-dot-com.com** (f

not-dot-com.com is a high-flying, fast-growing, big perfect choice for your retirement portfolio!

Click here for more information.

not-dot-com.com - Microsoft Internet Explorer

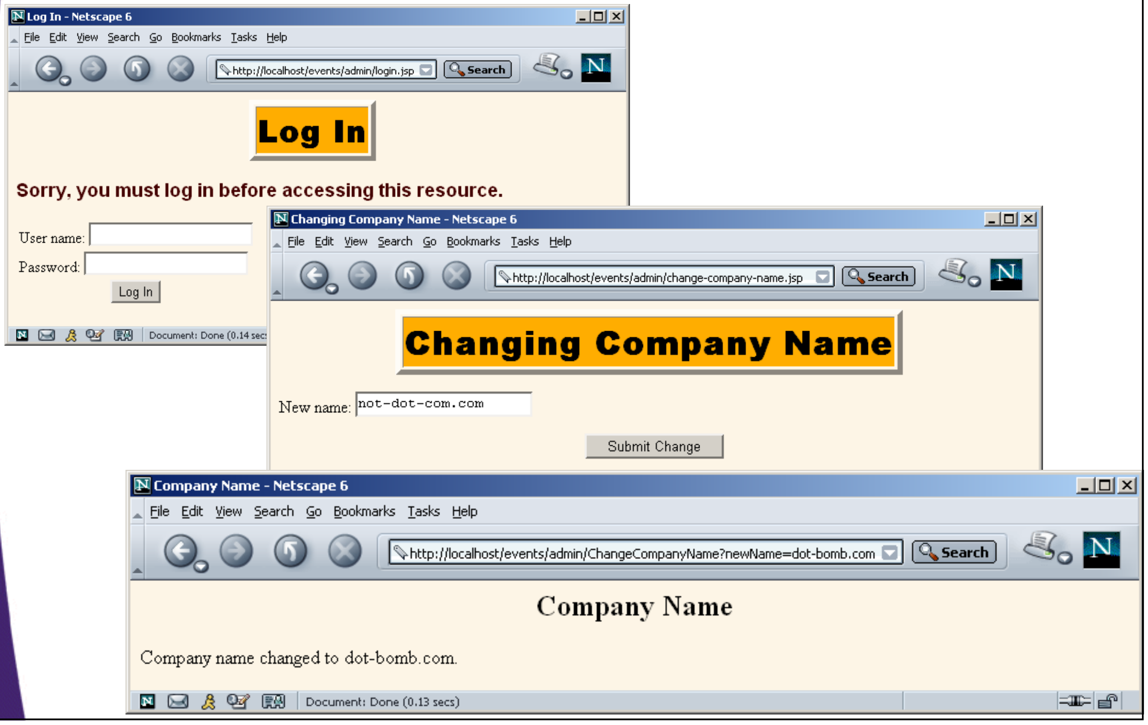
http://localhost/events/company-info.jsp

not-dot-com.com (formerly hot-dot-com.com)

Learn more about **not-dot-com.com** (formerly hot-dot-com.com)

- not-dot-com.com products
- not-dot-com.com services
- not-dot-com.com history
- investing in not-dot-com.com
- contacting not-dot-com.com

Example: Updating Former Company Name (Results)



Log In - Netscape 6

http://localhost/events/admin/login.jsp

Log In

Sorry, you must log in before accessing this resource.

User name:

Password:

Log In

Document: Done (0.14 sec

Changing Company Name - Netscape 6

http://localhost/events/admin/change-company-name.jsp

Changing Company Name

New name: not-dot-com.com

Submit Change

Company Name - Netscape 6

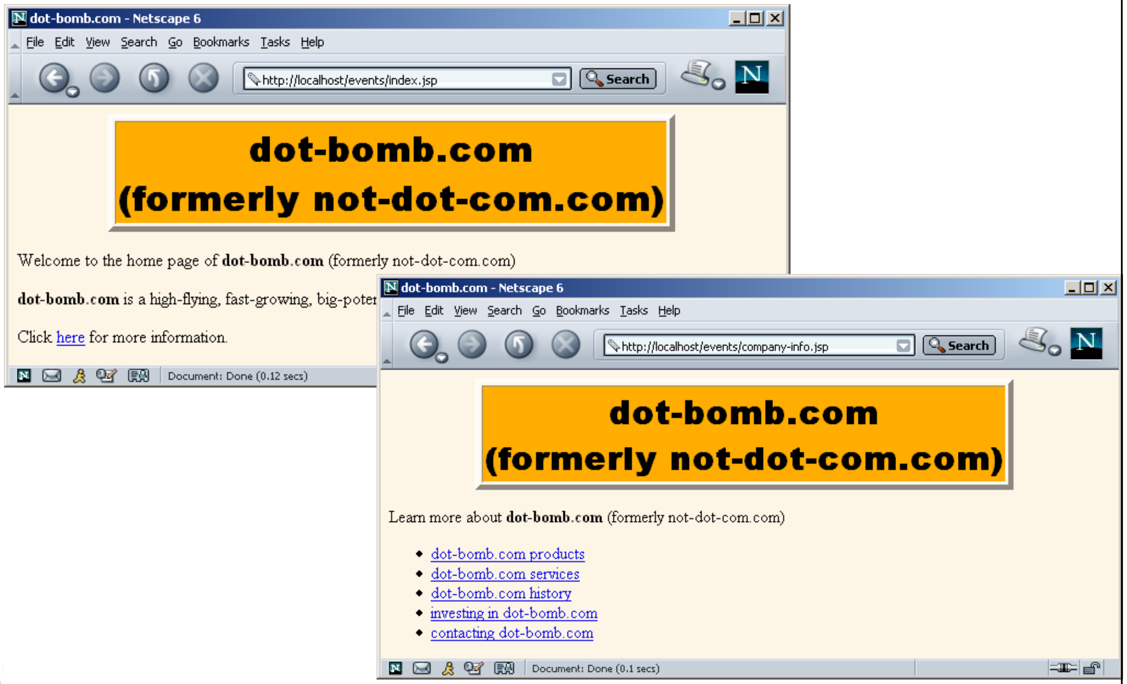
http://localhost/events/admin/ChangeCompanyName?newName=dot-bomb.com

Company Name

Company name changed to dot-bomb.com.

Document: Done (0.13 secs)

Example: Updating Former Company Name (Results)


dot-bomb.com - Netscape 6

http://localhost/events/index.jsp

dot-bomb.com (formerly not-dot-com.com)

Welcome to the home page of **dot-bomb.com** (formerly not-dot-com.com)

dot-bomb.com is a high-flying, fast-growing, big-poter

Click here for more information.

Document: Done (0.12 secs)

dot-bomb.com - Netscape 6

http://localhost/events/company-info.jsp

dot-bomb.com (formerly not-dot-com.com)

Learn more about **dot-bomb.com** (formerly not-dot-com.com)

- dot-bomb.com products
- dot-bomb.com services
- dot-bomb.com history
- investing in dot-bomb.com
- contacting dot-bomb.com

Document: Done (0.1 secs)

Packaging Listeners with Tag Libraries

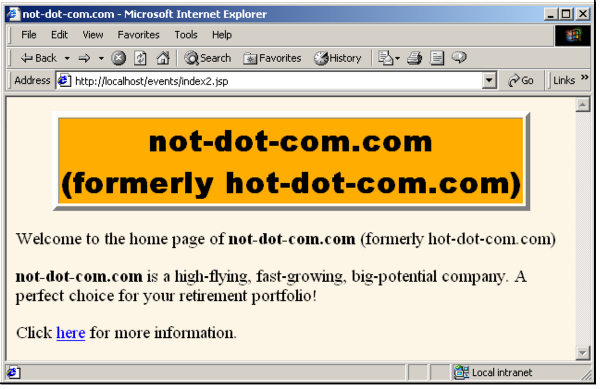
- **Tag libraries provide encapsulation of logic**
  - But what if tag depends on a listener?
    - E.g., a tag that uses servlet context attributes put there by a ServletContextListener
  - You'd like to include the listener with the tag library
    - Make it easy to reuse tag library in multiple Web apps
- **Generic approach**
  - Put tag library and listener classes in a JAR file
  - Drop JAR file in WEB-INF/lib
  - Put TLD file in WEB-INF
  - Requires you to modify WEB-INF/web.xml to declare listeners
    - Listener not purely associated with tag library

Packaging Listeners with Tag Libraries

- **New JSP 1.2 Behavior**
  - When Web application is loaded, the server *automatically* looks for .tld files in WEB-INF and its subdirectories
    - Not counting classes and lib directories
- **Approach to bundling listeners**
  - Move listener element from web.xml to the TLD file
  - Put TLD file in the WEB-INF hierarchy
- **Tomcat 4.0 problem**
  - Tomcat 4.0.x automatically detects TLD files only when the web.xml file contains `taglib` entries that gives an alias for the location of the TLD files. Works on other servers and Tomcat 4.1 and later – this is an implementation bug.
  - Defeats the purpose
    - If you have to modify web.xml, why not just declare listeners?

Packaging Listeners with Tag Libraries: Example

```
<%@ taglib uri="/company-name-taglib.tld"
           prefix="msajsp" %>
Learn more about <B><msajsp:companyName/></B>
<msajsp:formerCompanyName fullDescription="true"/>
<UL>
  <LI><A HREF="products.jsp"><msajsp:companyName/> products</A>
  <LI><A HREF="services.jsp"><msajsp:companyName/> services</A>
  <LI><A HREF="history.jsp"><msajsp:companyName/> history</A>
  <LI><A HREF="invest.jsp">investing in
      <msajsp:companyName/></A>
  <LI><A HREF="contact.jsp">
      contacting
      <msajsp:companyName/></A>
</UL>
```


not-dot-com.com - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://localhost/events/index2.jsp
Go Links
Local intranet

not-dot-com.com
(formerly hot-dot-com.com)

Welcome to the home page of **not-dot-com.com** (formerly hot-dot-com.com)

not-dot-com.com is a high-flying, fast-growing, big-potential company. A perfect choice for your retirement portfolio!

Click here for more information.

Packaging Listeners with Tag Libraries: Example TLD File

```
<taglib>…
  <listener>
    <listener-class>
     moreservlets.listeners.InitialCompanyNameListener
    </listener-class>
  </listener>
  <listener>
    <listener-class>
     moreservlets.listeners.ChangedCompanyNameListener
    </listener-class>
  </listener>
  <tag>
    <name>companyName</name>
    <tag-class>moreservlets.tags.CompanyNameTag
    </tag-class>
    <body-content>empty</body-content>
    <description>The current company name
    </description>
  </tag>  … </taglib>
```

Packaging Listeners with Tag Libraries: web.xml <u>for Tomcat 4.0</u>

```
…
<web-app>
  …
  <taglib>
    <taglib-uri>
      /company-name-taglib.tld
    </taglib-uri>
    <taglib-location>
      /WEB-INF/company-name-taglib.tld
    </taglib-location>
  </taglib>
  …
</web-app>
```

Recognizing Session Creation and Destruction

- **The previous listeners monitor Web-application-wide data.**
  - What if you want to monitor user-specific data?
- **Session usage can be one of the main memory drains on your server**
  - How do you monitor how many sessions are in use?
- **The answer to both is the same**
  - Use session listeners!

Using HttpSessionListener

1. **Implement the HttpSessionListener interface.**
2. **Override sessionCreated and sessionDestroyed.**
   - sessionCreated is triggered when a new session is created.
   - sessionDestroyed is triggered when a a session is destroyed. This destruction could be due to an explicit call to the invalidate method or because the elapsed time since the last client access exceeds the session timeout.
   - Multithreaded access is possible. Synchronize if necessary.
3. **Obtain a reference to the session and possibly to the servlet context.**
   - Each of the two HttpSessionListener methods takes an HttpSessionEvent as an argument. The HttpSessionEvent class has a getSession method that provides access to the session object. You almost always want this reference; you occasionally also want a reference to the servlet context. If so, first obtain the session object and then call getServletContext on it.

Using HttpSessionListener

- **Use the objects.**
  - One of the only methods you usually call on the session is setAttribute. Do this in sessionCreated if you want to guarantee that all sessions have a certain attribute.
  - Wait! What about getAttribute? Nope. In sessionCreated, there is nothing in the session yet, so getAttribute is pointless. In addition, all attributes are removed before sessionDestroyed is called, so calling getAttribute is also pointless there. If you want to clean up attributes that are left in sessions that time out, you use the attributeRemoved method of HttpSessionAttributeListener. So, sessionDestroyed is mostly reserved for listeners that are simply keeping track of the number of sessions in use.
- **Declare the listener.**
  - In web.xml or the TLD file, use `listener` and `listener-class` to list fully qualified name of listener class, as below.

```
<listener>
  <listener-class>package.SomeListener</listener-class>
</listener>
```

Example: Tracking Active Sessions

```
public class SessionCounter
    implements HttpSessionListener {
  private int totalSessionCount = 0;
  private int currentSessionCount = 0;
  private int maxSessionCount = 0;
  private ServletContext context = null;

  public void sessionCreated(HttpSessionEvent event) {
    totalSessionCount++;
    currentSessionCount++;
    if (currentSessionCount > maxSessionCount) {
      maxSessionCount = currentSessionCount;
    }
    if (context == null) {
      storeInServletContext(event);
    }
  }
```

Example: Tracking Active Sessions (Continued)

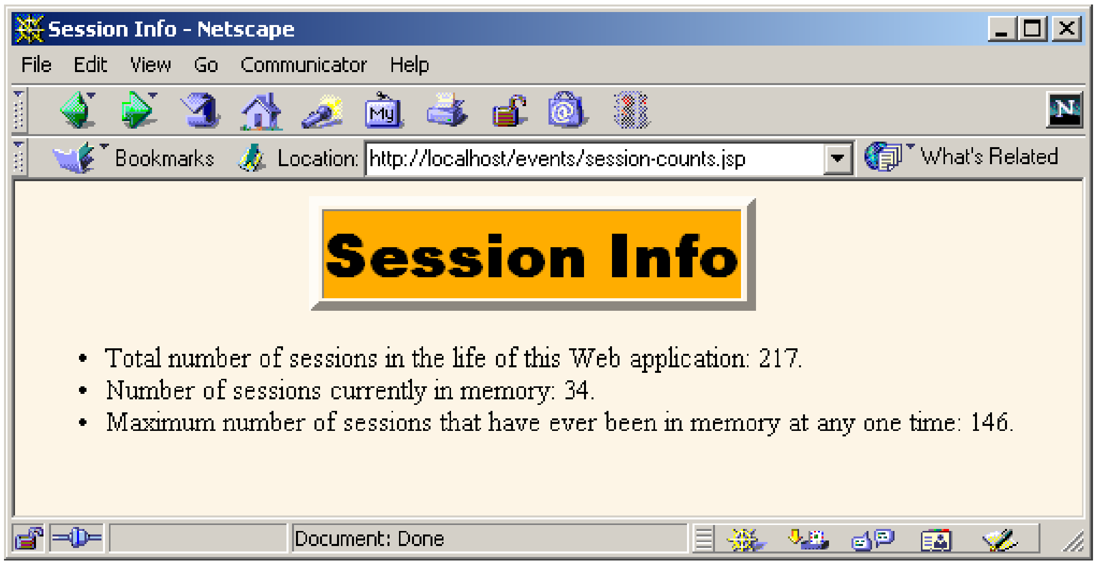
```
public void sessionDestroyed(HttpSessionEvent event){
  currentSessionCount--;
}
…

// Register self in the servlet context so that
// servlets and JSP pages can access the session
// counts.

private void storeInServletContext(HttpSessionEvent
event) {
  HttpSession session = event.getSession();
  context = session.getServletContext();
  context.setAttribute("sessionCounter", this);
}
}
```

Problem: How to Test Session Usage?

- Disable cookies, then use frame-based page that accesses lots of pages at once:

```
<FRAMESET ROWS="*,*,*,*" COLS="*,*,*,*">
  <FRAME SRC="test.jsp"><FRAME SRC="test.jsp">
  <FRAME SRC="test.jsp"><FRAME SRC="test.jsp">
  <FRAME SRC="test.jsp"><FRAME SRC="test.jsp">
  <FRAME SRC="test.jsp"><FRAME SRC="test.jsp">
  <FRAME SRC="test.jsp"><FRAME SRC="test.jsp">
  <FRAME SRC="test.jsp"><FRAME SRC="test.jsp">
  <FRAME SRC="test.jsp"><FRAME SRC="test.jsp">
  <FRAME SRC="test.jsp"><FRAME SRC="test.jsp">
  <NOFRAMES><BODY>
    This example requires a
    frame-capable browser.
  </BODY></NOFRAMES>
</FRAMESET>
```

Example: Tracking Active Sessions (Results)


Session Info - Netscape
File Edit View Go Communicator Help
Bookmarks
Location: http://localhost/events/session-counts.jsp
What's Related
Document: Done

Session Info

- Total number of sessions in the life of this Web application: 217.
- Number of sessions currently in memory: 34.
- Maximum number of sessions that have ever been in memory at any one time: 146.

Watching for Changes in Session Attributes

- **HttpSessionListener uses:**
  – Lets you detect when a session is created or destroyed.
- **HttpSessionListener deficiencies:**
  – Since session attributes are removed before session destruction, this listener does not let you clean up attributes that are in destroyed sessions.
- **Solution:**
  – HttpSessionAttributeListener interface.

Using HttpSessionAttributeListener

1. **Implement HttpSessionAttributeListener.**
2. **Override attributeAdded, attributeReplaced, and attributeRemoved.**
   - attributeAdded is triggered when a new attribute name is first added to a session.
   - attributeReplaced is triggered when a new value is assigned to an existing name. attributeAdded is *not* triggered in this case. The old value is obtained via event.getValue and the new value is obtained via session.getAttribute.
   - attributeRemoved is triggered when a session attribute is removed altogether. This removal can be due to an explicit programmer call to removeAttribute, but is more commonly due to the system removing all attributes of sessions that are about to be deleted because their timeout expired.

Using HttpSessionAttributeListener

3. **Obtain references to the attribute name, attribute value, session, and servlet context.**
   - The HttpSessionAttributeListener methods take an HttpSessionBindingEvent as args. HttpSessionBindingEvent has three useful methods: getName (name of attribute that was changed), getValue (value of changed attribute—new value for attributeAdded and previous value for attributeReplaced and attributeRemoved), and getSession (the HttpSession object). If you want access to the servlet context, first obtain the session and then call getServletContext on it.
4. **Use the objects.**
   - The attribute name is usually compared to a stored name to see if it is the one you are monitoring. The attribute value is used in an application-specific manner. The session is usually used to read previously stored attributes (getAttribute) or to store new or changed attributes (setAttribute).
5. **Declare the listener.**
   - Use `listener` and `listener-class` in *web.xml* as before.

Example: Monitoring Yacht Orders

```
public class YachtWatcher
    implements HttpSessionAttributeListener {

  private String orderAttributeName
    = "orderedItem";
  private String purchaseAttributeName
    = "purchasedItem";
  private String itemName = "yacht";
```

Example: Monitoring Yacht Orders (Continued)

```
/** Checks for initial ordering and final purchase of
 *  yacht. Records "Customer ordered a yacht" if the
 *  orderedItem attribute matches "yacht".
 *  Records "Customer finalized purchase of a yacht" if the
 *  purchasedItem attribute matches "yacht".
 */

public void attributeAdded(HttpSessionBindingEvent event){
  checkAttribute(event, orderAttributeName, itemName,
                 " ordered a ");
  checkAttribute(event, purchaseAttributeName, itemName,
                 " finalized purchase of a ");
}
```

Example: Monitoring Yacht Orders (Continued)

```
/** Checks for order cancellation: was an order for "yacht"
 *  cancelled?  Records "Customer cancelled an order for
 *  a yacht" if the orderedItem attribute matches "yacht".
 */

public void attributeRemoved(HttpSessionBindingEvent event)
{
  checkAttribute(event, orderAttributeName, itemName,
                 " cancelled an order for a ");
}
```

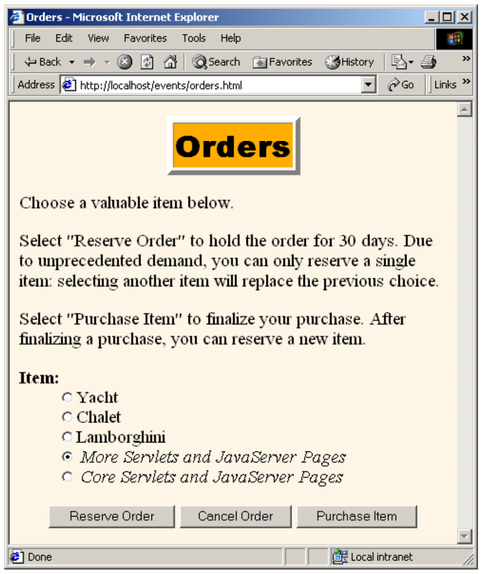
Example: Monitoring Yacht Orders (Continued)

```
/** Checks for item replacement: was "yacht" replaced
 *  by some other item? Records "Customer changed to a new
 *  item instead of a yacht" if the orderedItem attribute
 *  matches "yacht".
 */

public void attributeReplaced
                        (HttpSessionBindingEvent event) {
  checkAttribute(event, orderAttributeName, itemName,
                 " changed to a new item instead of a ");
}
```

Example: Monitoring Yacht Orders (Continued)

```
private void checkAttribute
                    (HttpSessionBindingEvent event,
                     String orderAttributeName,
                     String keyItemName,
                     String message) {
  String currentAttributeName = event.getName();
  String currentItemName = (String)event.getValue();
  if (currentAttributeName.equals(orderAttributeName)
      && currentItemName.equals(keyItemName)) {
    ServletContext context =
      event.getSession().getServletContext();
    context.log("Customer" + message +
                keyItemName + ".");
  }
 }
}
```

Example: Monitoring Yacht Orders (Results)


Orders - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://localhost/events/orders.html
Go Links

Orders

Choose a valuable item below.

Select "Reserve Order" to hold the order for 30 days. Due to unprecedented demand, you can only reserve a single item: selecting another item will replace the previous choice.

Select "Purchase Item" to finalize your purchase. After finalizing a purchase, you can reserve a new item.

Item:
- ○ Yacht
- ○ Chalet
- ○ Lamborghini
- ● *More Servlets and JavaServer Pages*
- ○ *Core Servlets and JavaServer Pages*

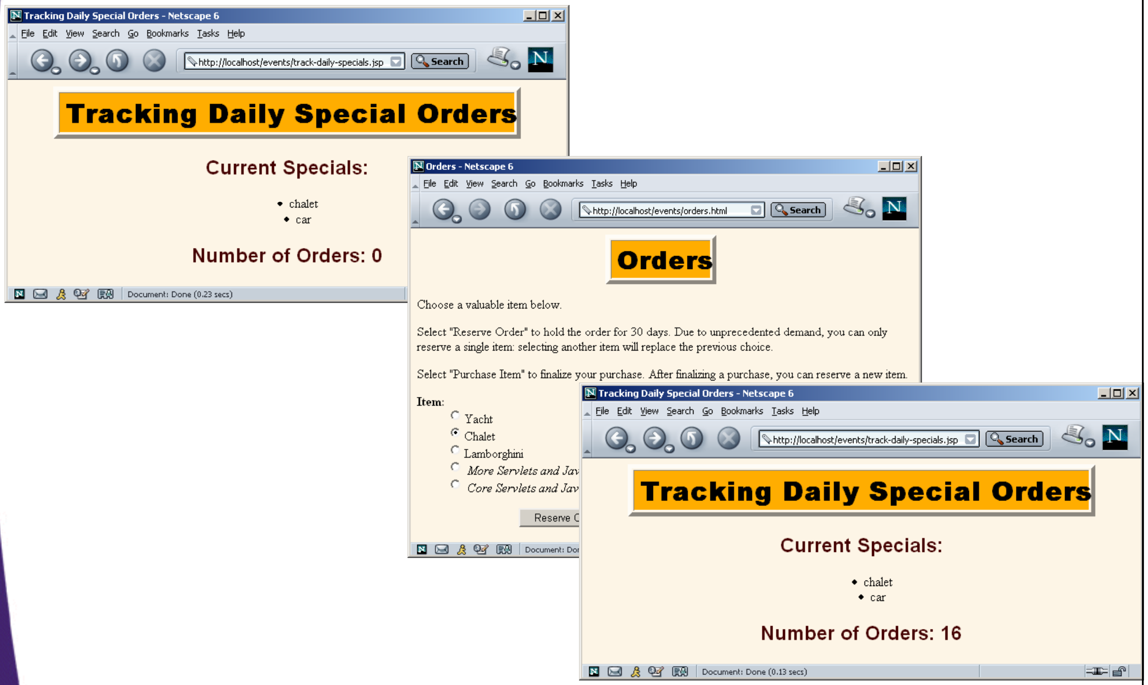
Log File

```
2001-11-07 11:50:59 Customer ordered a yacht.
2001-11-07 11:51:06 Customer changed to a new item instead of a yacht.
2001-11-07 11:52:37 Customer cancelled an order for a yacht.
2001-11-07 11:53:05 Customer finalized purchase of a yacht.
2001-11-07 11:53:35 Customer ordered a yacht.
2001-11-07 11:53:50 Customer cancelled an order for a yacht.
2001-11-07 11:54:20 Customer changed to a new item instead of a yacht.
2001-11-07 11:54:27 Customer changed to a new item instead of a yacht.
2001-11-07 11:54:42 Customer cancelled an order for a yacht.
2001-11-07 11:54:44 Customer ordered a yacht.
2001-11-07 11:54:47 Customer changed to a new item instead of a yacht.
```

Using Multiple Cooperating Listeners

- **Some tasks cannot be accomplished with any one listener type**
  - A combination is required
- **For example, you want to monitor purchases of the current daily special**
  - ServletContextListener to set up application-wide information about the session attributes that store daily specials
  - ServletContextAttributeListener to monitor changes to the attributes that store the information
  - HttpSessionAttributeListener to keep a running count of orders for the daily special.

Using Multiple Cooperating Listeners (Results)


Tracking Daily Special Orders - Netscape 6
File Edit View Search Go Bookmarks Tasks Help
http://localhost/events/track-daily-specials.jsp
Search
Document: Done (0.23 secs)
Orders - Netscape 6
http://localhost/events/orders.html
Reserve
Document: Do
Document: Done (0.13 secs)

Tracking Daily Special Orders

Current Specials:

- chalet
- car

Number of Orders: 0

Orders

Choose a valuable item below.

Select "Reserve Order" to hold the order for 30 days. Due to unprecedented demand, you can only reserve a single item: selecting another item will replace the previous choice.

Select "Purchase Item" to finalize your purchase. After finalizing a purchase, you can reserve a new item.

Item:

- Yacht
- [✓ Chalet]
- Lamborghini
- *More Servlets and Jav*
- *Core Servlets and Jav*

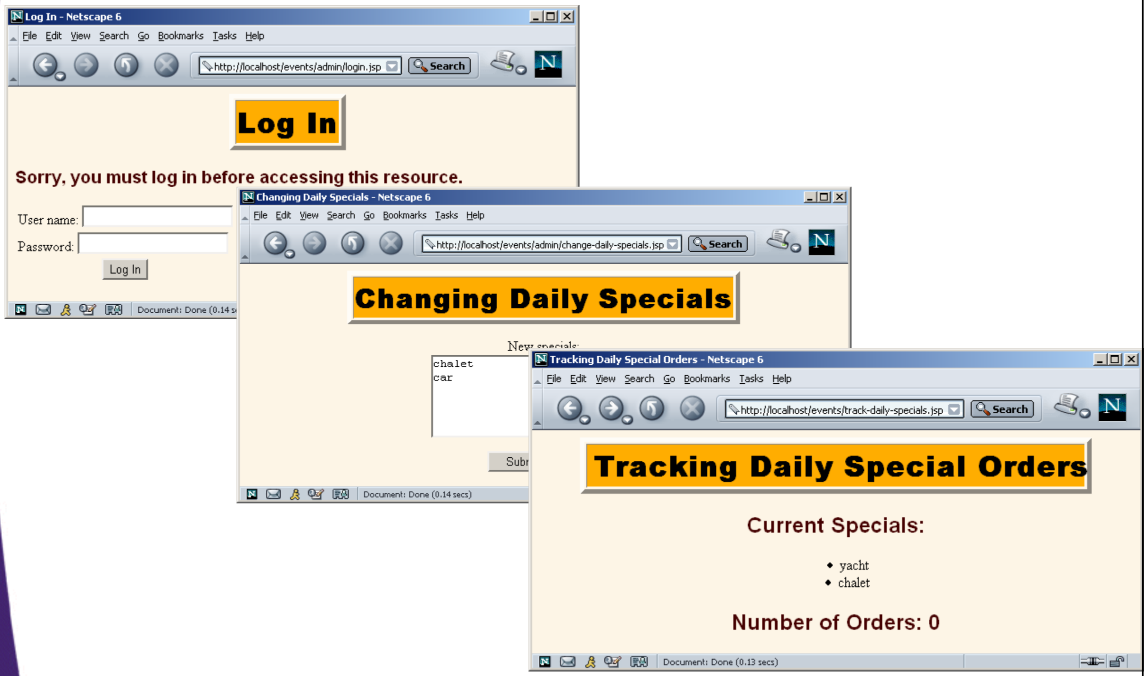
Tracking Daily Special Orders

Current Specials:

- chalet
- car

Number of Orders: 16

Using Multiple Cooperating Listeners (Results Continued)


Log In - Netscape 6
http://localhost/events/admin/login.jsp
Changing Daily Specials - Netscape 6
http://localhost/events/admin/change-daily-specials.jsp
Tracking Daily Special Orders - Netscape 6
http://localhost/events/track-daily-specials.jsp

Log In

Sorry, you must log in before accessing this resource.

User name:

Password:

Log In

Changing Daily Specials

New specials:

chalet
car

Tracking Daily Special Orders

Current Specials:

- yacht
- chalet

Number of Orders: 0

Summary

- **Available listeners**
  - Servlet context listeners.
    - Notified when servlet context is initialized and destroyed.
  - Servlet context attribute listeners.
    - Notified when context attributes are added/removed/replaced
  - Session listeners.
    - Notified when sessions are created, invalidated, or timed out.
  - Session attribute listeners.
    - Notified when session attributes are added/removed/replaced
- **Steps**
  - Implement interface, override methods, access the

Questions?