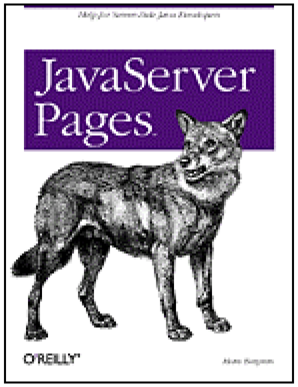
JavaServer Pages (JSP) Custom Actions

Georgia Web Developers Conference, July 25, 2001

Who I Am

- Hans Bergsten
  - hans@gefionsoftware.com
  - President of Gefion software
  - Member of the JSP working group (JCP)
  - 

Author of *JavaServer Pages* (O'Reilly)

Overview

- The JSP promise
- Custom Action Components
- Implementation Examples
- New in JSP 1.2
- Q & A

Not a JSP introduction! Plenty of Code!

Before JSP

```
public void doGet(…) {
  ResultSet rs = getCustomers(“VIP”);
  out.println(“<ul>”);
  while (rs.next()) {
    out.println(“<li>” + rs.getString(“Name”);
  }
```

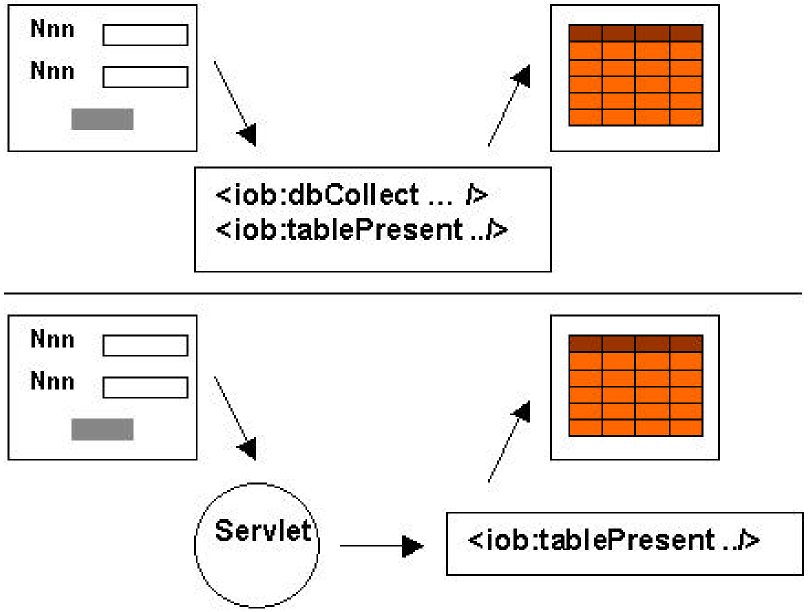
With JSP 1.0

```
<jsp:useBean id=“cdb” class=“CustDB” />
<% ResultSet rs =
      cdb.getCustomers(“VIP”); %>
<ul>
<%  while (rs.next()) { %>
        <li> <%= rs.getString(“Name”) %>
<%  } %>
```

With JSP 1.1 Custom Actions

```
<foo:getVIPCust id=“vips” />
<ul>
<foo:loop name=“vips” loopId=“cust” >
   <li> <jsp:getProperty name=“cust”
              property=“name” />
</foo:loop>
</ul>
```

Types of Custom Actions



- JSP for all MVC roles
  - Database access
  - EJB access
  - Input Validation
- JSP just as View
  - Presentation
  - Conditional processing

Custom Action Representation

```
<foo:loop name=“vips”
  loopId=“cust”>
  <li> …
</foo:loop>
```

or

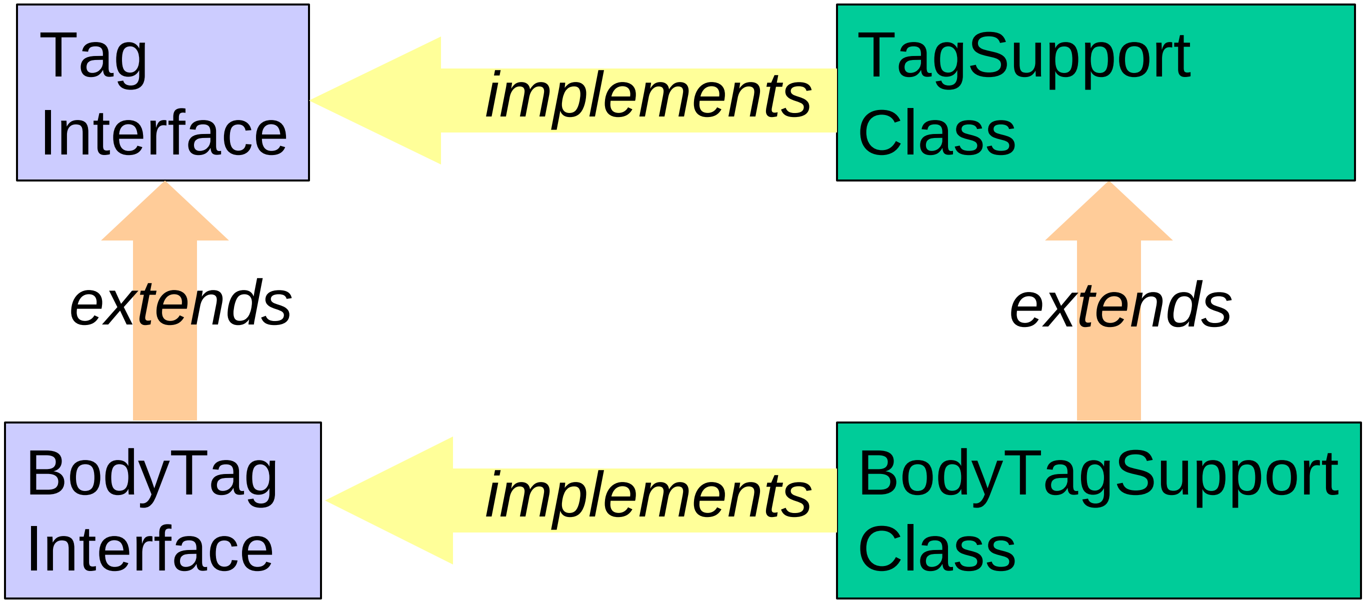
```
<foo:getVIPCust
  id=“vips” />
```

- XML element
  - Opening tag
  - Body
  - Closing tag
  - Shorthand notation for element without a body

Custom Action Components

- Tag handler class
- Tag Library Descriptor (TLD)
- TagExtraInfo subclass

Interfaces and Support Classes



Basic Tag Handler

```
<foo:ifEq param=“sale”
  value=“true”>
  <b>On Sale!</b>
</foo:ifEq>
```

```
XXX implements Tag {
  setPageContext(…)
  setParam(...)
  setValue(…)
  doStartTag()


  doEndTag()
```

PageContext Class

- Provides access to all context objects
  - getRequest() and getResponse()
  - getSession() and getServletContext()
- Provides access to JSP scope objects
  - setAttribute(), getAttribute(), removeAttribute()
  - findAttribute()

Implementing the IfEqTag

```
public class IfEqTag extends TagSupport {
  …
  public int doStartTag() {
    ServletRequest req = pageContext.getRequest();
    String paramValue = req.getParameter(param);
    if (value.equals(paramValue)) {
      return EVAL_BODY_INCLUDE;
    else {
      return SKIP_BODY;
  }
}
```

Tag Library Descriptor (TLD)

```
<taglib>
  <tlibversion>1.0</tlibversion>
  <jspversion>1.1</jspversion>
  <shortname>foo</shortname>
  <uri>/foolib</uri>
  <info>Sample Tag Library</info>
  ...
```

Tag Library Descriptor (TLD)

```
…
<tag>
  <name>ifEq</name>
  <tagclass>com.foo.IfEqTag</tagclass>
  <teiclass>com.foo.IfEqTEI</teiclass>
  <bodycontent>JSP</bodycontent>
  ...
```

Tag Library Descriptor (TLD)

```
    <attribute>
      <name>param</name>
      <required>true</required>
      <rtexprvalue>true</rtexprvalue>
    </attribute>
  </tag>
</taglib>
```

Tag Library Deployment

index.jsp

images/logo.gif

WEB-INF/classes/com/foo/IfEqTag.class

WEB-INF/tlds/footaglib.tld

WEB-INF/lib/footaglib.jar

WEB-INF/web.xml

...

web.xml Declaration

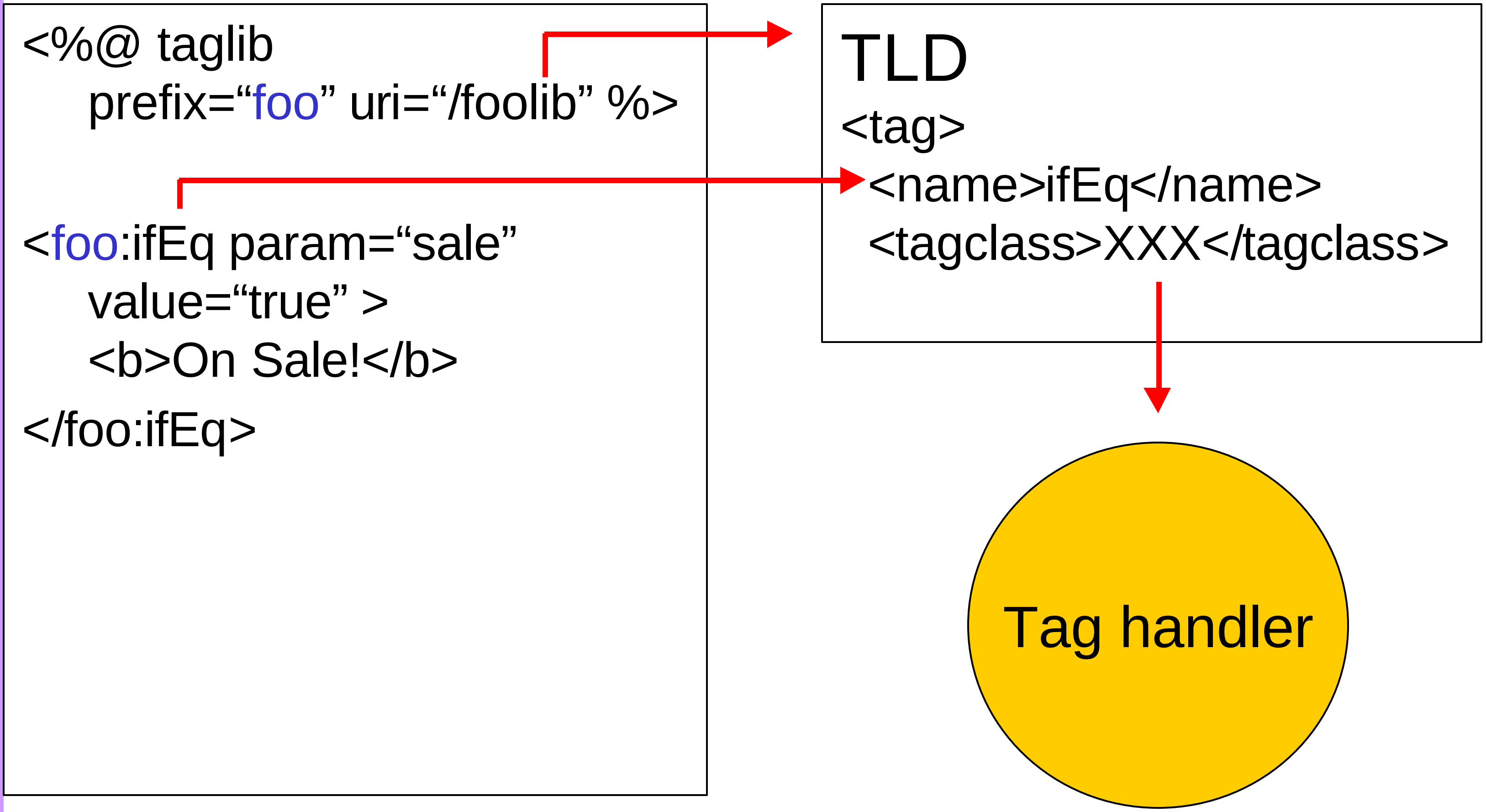
```
<web-app>
  …
  <taglib>
    <taglib-uri>/foolib</taglib-uri>
    <taglib-location>
      /WEB-INF/tlds/foolib.tld
    </taglib-location>
```

JSP taglib Directive

```
<%@ taglib uri=“/foolib” prefix=“foo” %>

<foo:ifEq param=“sale” value=“true”>
  <b>On Sale!</b>
</foo:ifEq>
```

How It All Fits Together



```
<%@ taglib
    prefix="foo" uri="/foolib" %>

<foo:ifEq param="sale"
    value="true" >
    <b>On Sale!</b>
</foo:ifEq>
```

TLD

```
<tag>
  <name>ifEq</name>
  <tagclass>XXX</tagclass>
```

Tag handler

Basic Tag Examples

- Conditional processing:
  - doStartTag() -> EVAL_BODY_INCLUDE, SKIP_BODY
  - doEndTag() -> EVAL_PAGE, SKIP_PAGE
- Presentation: i18n, complex tables, etc.
- Data access: database, EJB, etc.

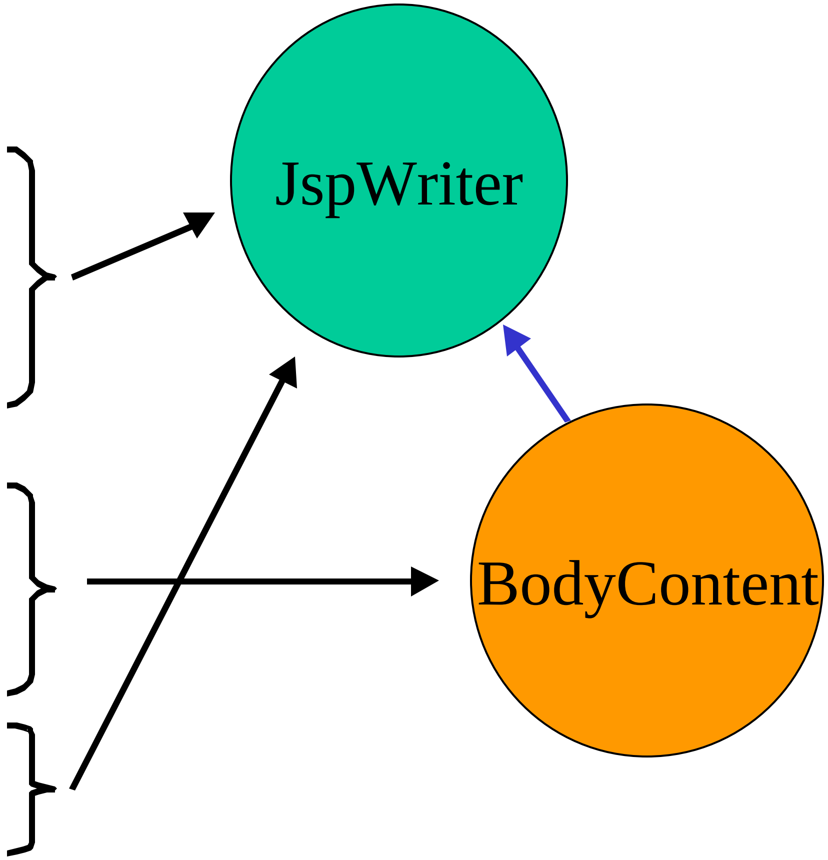
Processing the Element Body

```
<foo:mail to="...">
  Dear
  <jsp:getProperty ...>
  ...
</foo:mail>
```

```
XXX impl. BodyTag {
  setPageContext(…)
  setTo(...)
  doStartTag()
  setBodyContent(...)
  doInitBody()
  doAfterBody()
  doEndTag()
```

Buffering Hierarchy

```
<%@ taglib … %>
<html>
  Some text
  <jsp:getProperty .../>
  <foo:mail …>
    Dear
      <jsp:getProperty />
  </foo:mail>
</html>
```



BodyContent class

- JspWriter subclass, buffers the element body evaluation result for BodyTags
  - getReader()
  - getString()
  - writeOut(java.io.Writer out)
  - getEnclosingWriter()

Implementing the MailTag

```
public class MailTag extends BodyTagSupport {
  …
  public int doAfterBody() {
    mailBean.setTo(to);
    mailBean.setMsg(getBodyContent().getString());
    mailBean.deliver();
    return SKIP_BODY;
  }
}
```

Iterating Over the Body

```
<foo:loop name=“vips” loopId=“cust”>
  <ul>
    <jsp:getProperty name=“curr”
      property=“name” />
</foo:loop>
```

Implementing the LoopTag

```
public class LoopTag extends BodyTagSupport {
  …
  private Iterator iterator;
  public int doStartTag() {
    // Set up an Iterator for the Collection
    Collection coll =
      (Collection) pageContext.findAttribute(name);
    iterator = coll.iterator();
```

Implementing the LoopTag

```
  if (iterator.hasNext()) {
    pageContext.setAttribute(loopId, iterator.next());
    return EVAL_BODY_TAG;
  }
  else {
    return SKIP_BODY;
  }
}
```

Implementing the LoopTag

```
public int doAfterBody() {
  if (iterator.hasNext()) {
    pageContext.setAttribute(loopId, iterator.next());
    return EVAL_BODY_TAG;
  }
  else {
    return SKIP_BODY;
  }
}
```

BodyTag Examples

- Dynamic input, using other JSP elements to create it
- Static input spanning multiple lines, e.g. SQL, XML, other non-JSP syntax.
- Iterations
  - doAfterBody() -> EVAL_BODY_TAG, SKIP_BODY

Validating Element Syntax

- Two mechanisms:
  - Container validation using the TLD:

    ```
    <attribute>
      <name>param</name>
      <required>true</required>
    </attribute>
    ```

  - Use a TagExtraInfo class with isValid()
- Can be combined

Validation with TagExtraInfo

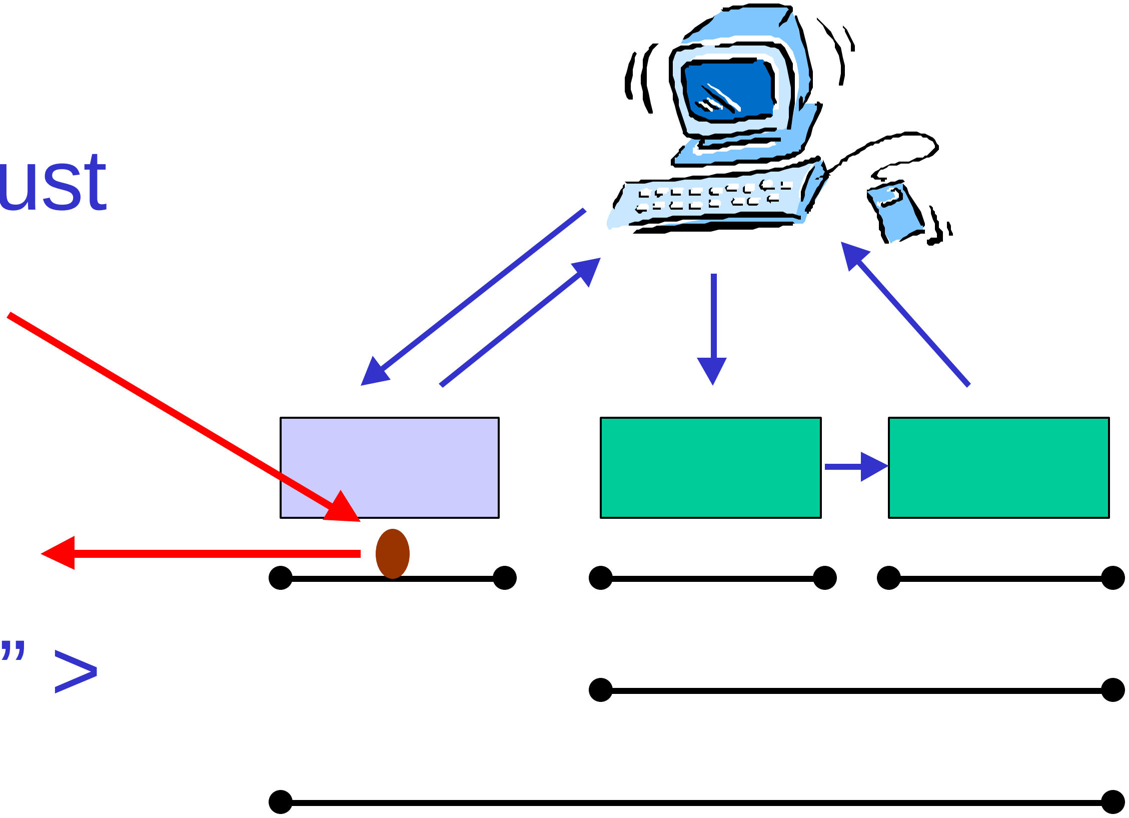
```
public class MailTEI extends TagExtraInfo {
  public boolean isValid(TagData data) {
    if (data.getAttribute(“to”) != null &&
        data.getAttribute(“toList”) != null) {
          return false; // Mutually exclusive optional
    }
    if (data.getAttribute(“name”) != null &&
        data.getAttribute(“property”) == null) {
        return false;  // Dependent optional
    }
    return true;
```

Letting Actions Cooperate

- Through objects in a JSP scope
- Through direct method calls on the parent tag handler

Coop Through Scoped Objects

```
<foo:getVIPCust
  id=“vips” />
<foo:loop
  name=“vips”
  loopId=“cust” >
```



Coop Through Scoped Objects

- Save the object in one tag handler:

```
public int doXXX() {
  ...
  pageContext.setAttribute(id, value, scope);
}
```

- Get the object in another tag handler:

```
public int doXXX() {
  …
  Object o = pageContext.findAttribute(name);
}
```

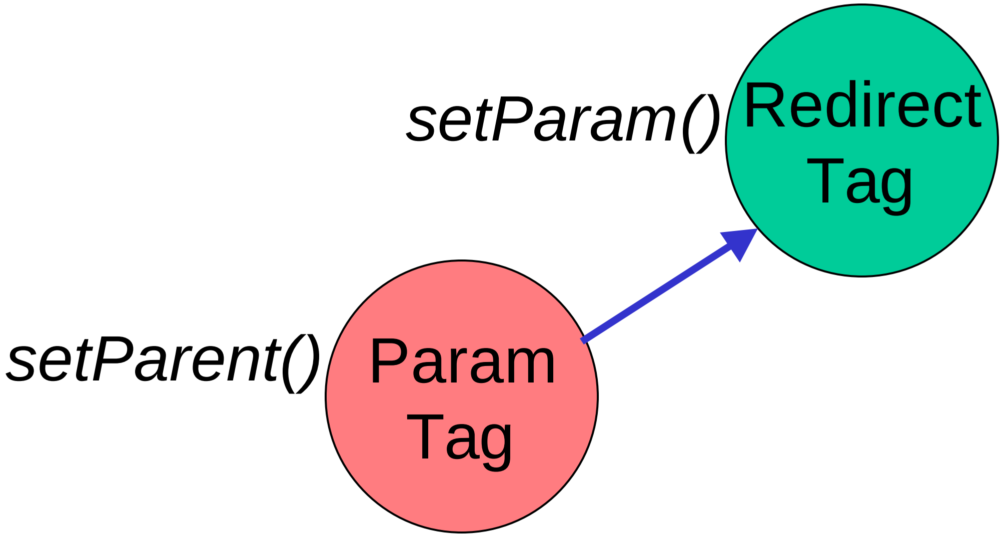
Scripting Variable Assignment

```
public class GetVIPCustTEI extends TagExtraInfo {
 public VariableInfo[] getVariableInfo(TagData data) {
  VariableInfo vi = new VariableInfo[1];
  vi[0] = new VariableInfo(
   data.getAttributeString(“id”), // Variable name
   “java.util.Collection”,        // Variable type
   true,                          // Declare variable?
   VariableInfo.AT_END);          // Assignment scope
  return vi;
 }
}
```

Scripting Variable Assignment

```
GetVIPTag t1 = new GetVIPTag();
t1.setPageContext(pageContext);
…
t1.doStartTag();
t1.doEndTag();
Collection vips = (Collection)
  pageContext.findAttribute(“vips”);
```

Coop Through Method Calls



```
<foo:redirect
  page="some.jsp">
  <foo:param name="a" value="b" />
</foo:redirect>
```

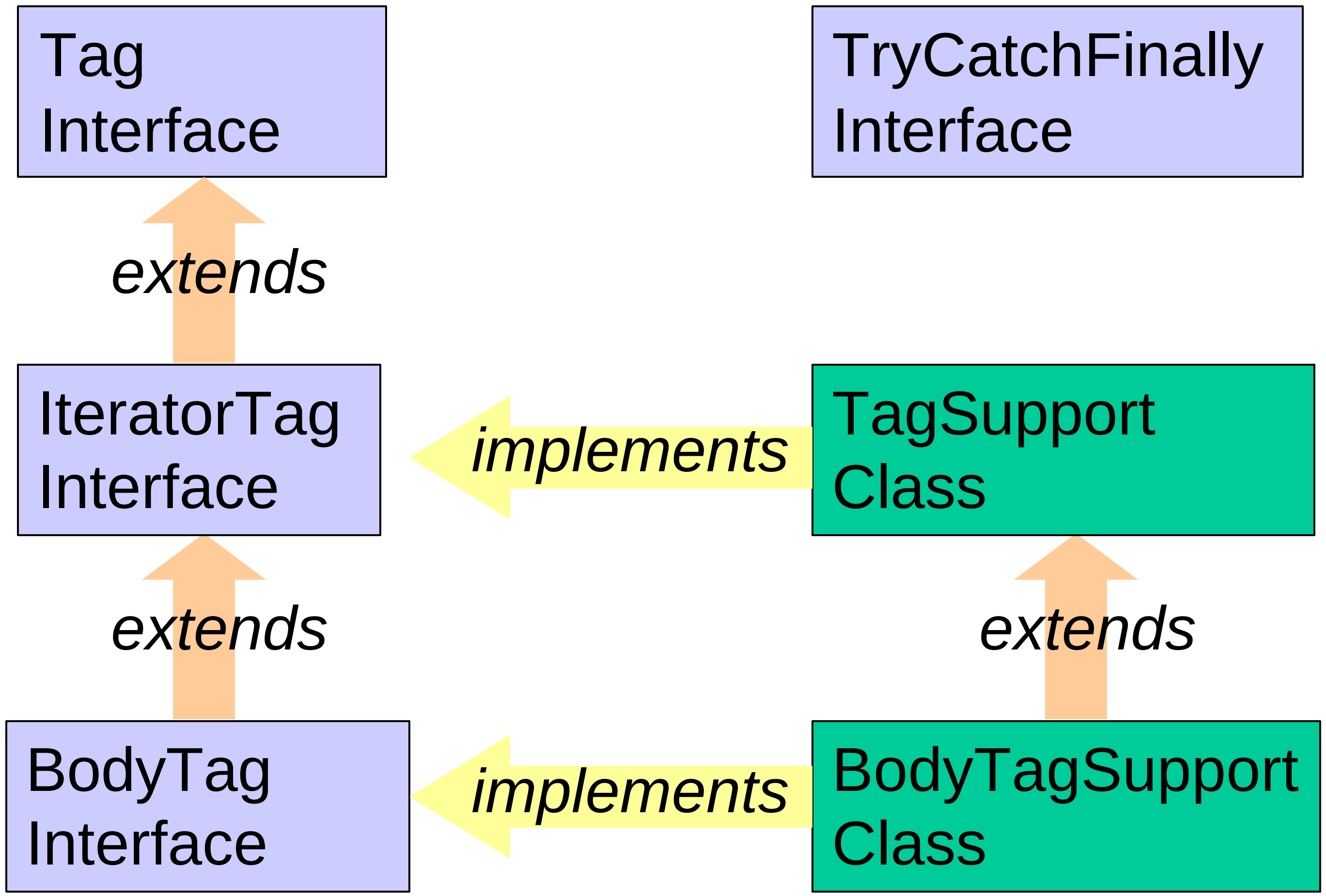
Coop Through Method Calls

```
public class ParamTag extends TagSupport {
  public int doEndTag() {
    RedirectTag parent = (RedirectTag)
      findAncestorWithClass(this, RedirectTag.class);
    if (parent != null) {
      parent.setParam(name,
        URLEncoder.encode(value);
    }
    return EVAL_PAGE;
  }
}
```

JSP 1.2 Tag Library News

- New Interfaces
  - IteratorTag, TryCatchFinally

New Interfaces



TryCatchFinally

- public void doCatch(Throwable t) throws Throwable

  Handle exceptions thrown in body or doXXX() method
- public void doFinally()

  Called even in case of exceptions in body or doXXX() methods

JSP 1.2 Tag Library News

- New Interfaces
  - IteratorTag, TryCatchFinally
- Validation Enhancements
  - TagLibraryValidator class
- Application & session event listeners
- TLD can describe tag handler variables

More Info

- JSP Specification
  http://java.sun.com/products/jsp/
- JavaServer Pages book
  http://TheJSPBook.com/
- Jakarta Taglibs & Struts
  http://jakarta.apache.org/
- InstantOnline Basic
  http://www.gefionsoftware.com/InstantOnline/