

# Shale: The Next Struts?

**Craig R. McClanahan**

Senior Staff Engineer  
Sun Microsystems, Inc.

<http://struts.apache.org/struts-shale/>



# Agenda

What is Shale?

Validation

Spring and Tiles Integration

Tapestry-like Views

Remoting

View Controller

Dialogs

Summary

# Agenda

## What is Shale?

Validation

Spring and Tiles Integration

Tapestry-like Views

Remoting

View Controller

Dialogs

Summary

# What Is Shale?

- An extension of JavaServer Faces technology
- Originally proposed as “Struts 2.0”
  - Accepted as a Struts sub-project
- No direct connection with Struts Action Framework
  - But provides corresponding functionality
- Not a repository of general purpose JavaServer Faces components

# Shale Tag Library

- `<s:token>`
- `<s:subview>`
- `<s:commonsValidator>`
- `<s:validatorScript>`
- `<s:clay>`

# Utilizes JSF Extension Points

- ShalePropertyResolver
- ShaleVariableResolver
- DialogNavigationHandler
- ShaleViewHandler
- TilesViewHandler

# DEMO

Shale at work

# Agenda

What is Shale?

**Validation**

Spring and Tiles Integration

Tapestry-like Views

Remoting

View Controller

Dialogs

Summary

# Shale Validation Concepts

- Utilizes *Jakarta Commons Validator*
  - Same functionality used by Struts Action Framework
- Provides server-side **and** client side validation
  - Based on a single configured set of rules
- Implemented as a standard JSF Validator
  - `<s:commonsValidator>` tag in JSP pages
- Additional tag to incorporate generated JavaScript validation functions
  - `<s:validatorScript>` in JSP pages

# Using Shale Validation

```
<h:form id="form" onclick="validateForm">
  ...
  <h:inputText id="creditCard">
    <s:commonsValidator type="required"
      arg="{messages['cc.required']}" client="true"/>
    <s:commonsvalidator type="creditCard"
      arg="{messages['cc.format']}" server="true"/>
  </h:inputText>
  ...
  <s:validatorScript functionName="validateForm"/>
  ...
</h:form>
```

# DEMO

Shale validation

# Agenda

What is Shale?

Validation

**Spring and Tiles Integration**

Tapestry-like Views

Remoting

View Controller

Dialogs

Summary

# Shale and Spring

- JavaServer Faces *Managed Beans Facility* provides basic “Dependency Injection” service:
  - Create beans on demand
  - Initialize properties from metadata
  - Store in specified scope
- Spring provides custom *VariableResolver*:
  - Is there a managed beans definition? Use it
  - Is there a Spring bean definition? Use it
- Evaluating value binding expressions can trigger Spring BeanFactory bean creation

# Shale and Tiles

- Enclose `<tiles:insert>` tags in subviews:

```
<s:subview id="loginPage">
    <tiles:insert definition="loginPage.jsp"/>
</s:subview>
```

- Specify Tiles as navigation destinations:

```
<navigation-rule>
    <navigation-case>
        <from-outcome>todo-list</from-outcome>
        <to-view-id>todo-list.tile</to-view-id>
    </navigation-case>
</navigation-rule>
```

- Uses *standalone Tiles* not dependent on Struts

# DEMO

Spring and Tiles

# Agenda

What is Shale?

Validation

Spring and Tiles Integration

**Tapestry-like Views**

Remoting

View Controller

Dialogs

Summary

# Retro Views

- Use plain HTML to define your views:
  - Graphic artists can work with HTML
  - HTML elements reference components
- Define components in XML document
- Tie HTML elements to component definitions with *jsfid* attribute

# Two Usage Modes

- Preview:
  - Used by graphic artists and developers
  - <http://localhost:8080/myapp/logon.html>
- Runtime:
  - Used by developers and end users
  - <http://localhost:8080/myapp/logon.faces>
- Facility also supports reuse of component trees:
  - On-the-fly attribute substitution
  - Customization of value binding expressions

# DEMO

Tapestry-like Views

# Agenda

What is Shale?

Validation

Spring and Tiles Integration

Tapestry-like Views

**Remoting**

View Controller

Dialogs

Summary

# Remoting

- Map incoming URL to processing logic:

```
<catalog name="remote">
  <chain      name="/getCities.remote">
    <command
      className="com.mycompany.myapp.LoadState"
      state="stateAbbreviation"/>
    <command
      className="com.mycompany.myapp.GetCities"/>
      state="stateAbbreviation"/>
  </chain>
</catalog>
```

- Construct response with Java code or JSP
- Useful for server-side processing of Ajax requests

# Agenda

What is Shale?

Validation

Spring and Tiles Integration

Tapestry-like Views

Remoting

**View Controller**

Dialogs

Summary

# View Controller

- JavaServer Faces developers are typically familiar with the standard *request processing lifecycle* of each request
- A simpler programming model would be if all coding were in response to events
  - Hollywood Principle: “don't call us, we'll call you”
- Implement *ViewController* interface to receive support for application lifecycle events
  - Uses only standard JavaServer Faces APIs
  - Portable to any runtime environment

# Application Lifecycle Events

- Application associates a *backing bean* with each JSP page corresponding to a view:
  - Backing bean can implement *ViewController*
  - (Optional) Components bound to concrete properties
  - Methods created for UI event handlers
- Shale does *application lifecycle* callbacks:
  - `init()` -- When a page bean is first created
  - `preprocess()` -- Before a form submit is processed
  - `prerender()` -- Before a view is rendered
  - `destroy()` -- After rendering, if `init()` was called

# Agenda

What is Shale?

Validation

Spring and Tiles Integration

Tapestry-like Views

Remoting

View Controller

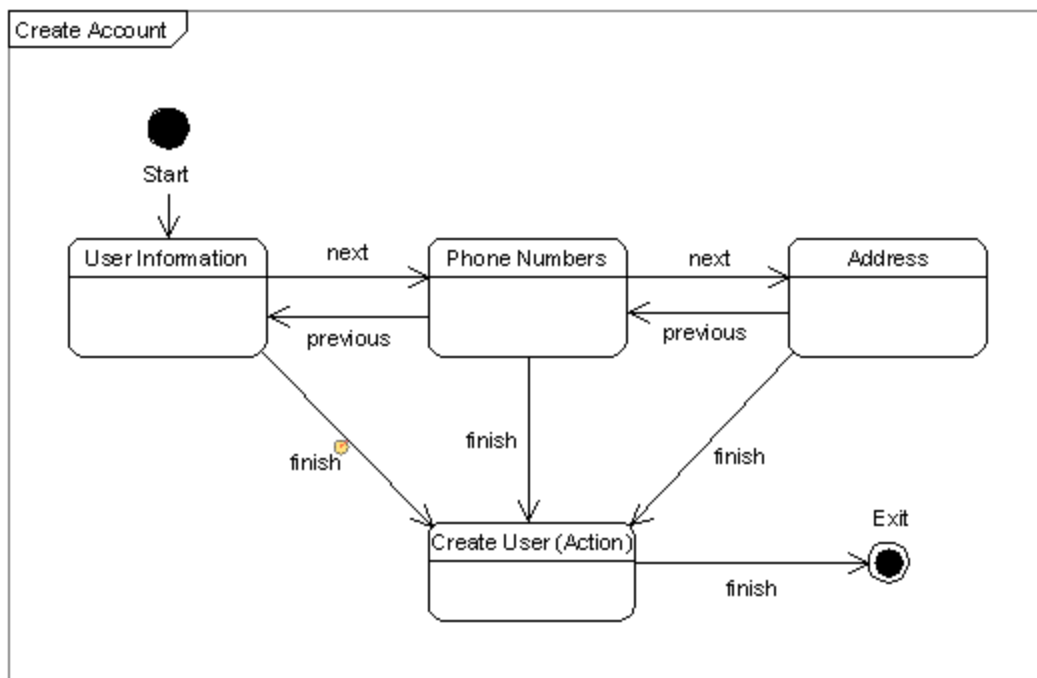
**Dialogs**

Summary

# Shale Dialog Concepts

- Based on design of *Spring Web Flow*:
  - Integrated with JavaServer Faces technology
  - Simpler, more intuitive API
- Models conversations as a *State Diagram*:
  - Action states
  - View states
  - Subdialog states
- Transitions between states based on *logical outcomes* returned by executing a state
- Supports “conversation scope” data storage

# Modelling Dialogs As State Diagrams



Created with Poseidon for UML Community Edition. Not for Commercial Use.

# Configuring Shale Dialogs

```
<dialogs>
  <dialog      name="Create Account"
              start="User Information">
    <view      name="User Information">
      <transition outcome="next" target="Phone Numbers"/>
      <transition outcome="finish"
                target="Create Account"/>
    </view>
    ...
    <action name="Create Account"
            method="{account.create}"/>
    <end      name="Exit" viewId="/login.jsp"/>
  </dialog>
</dialogs>
```

# Invoking Shale Dialogs

- Enter dialogs with a special logical outcome:
  - `<h:commandButton action="dialog:CreateAccount"/>`
- When within a dialog:
  - Custom NavigationHandler manages navigations
  - Per-dialog state information maintained
  - Subdialogs cause stack push/pop of state information
- When not within a dialog:
  - Standard JavaServer Faces navigation processing

# DEMO

## Shale Dialogs

# Agenda

What is Shale?

Validation

Spring and Tiles Integration

Tapestry-like Views

Remoting

View Controller

Dialogs

**Summary**

# Current Status of Shale

- Test Build 1.0.0 available:
  - <http://cvs.apache.org/dist/struts/shale/v1.0.0/>
- Other resources:
  - <http://struts.apache.org/struts-shale/>
  - <http://wiki.apache.org/struts/StrutsShale/>
  - <http://cvs.apache.org/builds/struts/nightly/struts-shale/>

# Future Directions

- Shale Tiger Extensions:
  - Layer on top, using JavaSE 5 annotations
  - Annotations for managed bean definition
  - Annotations for view controller callbacks
- Improve Dialogs Functionality:
  - Multiple simultaneously executing dialogs
  - Start dialogs programmatically
- Improve Clay Functionality:
  - Load multiple templates from one resource
  - More implicit decorators

# Future Directions

- Improve View Controller Functionality:
  - More robust exception handling support
  - Optional phase listener callbacks
- Improve Remoting Functionality:
  - Support JSF ResponseWriter for creating response
- What else would you like to see?

# Q&A

Craig R. McClanahan

# Shale: The Next Struts?

**Craig R. McClanahan**

Senior Staff Engineer  
Sun Microsystems, Inc.

<http://struts.apache.org/struts-shale/>

