

# Rich Enterprise Applications

Powered by Oracle Application  
Development Framework  
(Oracle ADF)



**ORACLE®**

## **Skinning and JavaScript in ADF Faces**

Frank Nimphius  
Senior Principal Product Manager

# Agenda : Skinning

- • About Skinning
- The Problem of directly added CSS
- How-to skin ADF Faces
- Skinning Support in JDeveloper
- Debugging
- Demo

# About Skinning

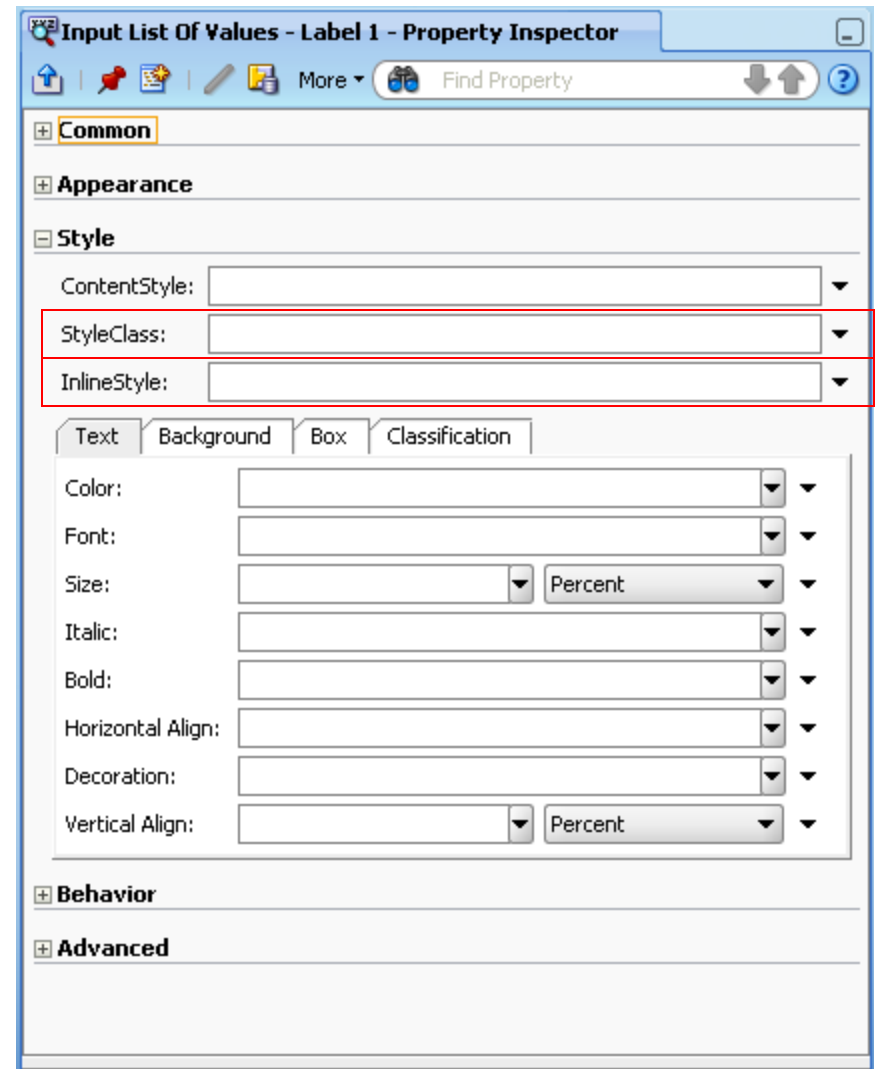
- A skin is a style sheet based on the CSS 3.0 syntax that is specified in one place for an entire application
- Developers can change the styles, icons, properties, and text of an ADF Faces component using skinning
- Skins use CSS to define the layout of ADF Faces and Trinidad components
  - Do not operate on HTML elements
  - Use component selectors
- Skins consist of
  - CSS file
  - Images
  - Localized strings

# Agenda : Skinning

- About Skinning
- • The Problem of directly added CSS
- How-to skin ADF Faces
- Skinning Support in JDeveloper
- Debugging
- Demo

# CSS in ADF Faces

- **StyleClass**
  - Can be used in conjunction with skinning
  - Applies styles to the root DOM element
  - Can use EL for context specific CSS
- **InlineStyle**
  - Styles the root DOM element
  - Can use EL for context specific CSS



# Agenda : Skinning

- About Skinning
- The Problem of directly added CSS
- • How-to skin ADF Faces
- Skinning Support in JDeveloper
- Debugging
- Demo

# How-to build Custom Skins

- Consult the skin-selectors.html page for all skinning keys defined for each component and global keys
- Creates a skinning .css file that uses the skinning keys to style the icons, properties, and styles
- The CSS file for your skin must be stored under the root of your Web application
- Make sure that all resources like images and other CSS files required for the skin are accessible

# How-to build Custom Skins

- Create a trinidad-skins.xml that gives the skin an id, family, render-kit-id, resource bundle
- Set the trinidad-config.xml's <skin-family> element value to the skin family name
- Place trinidad-skins.xml into WEB-INF directory or a JAR file's META-INF directory

# Configuration in trinidad-config.xml

```
<trinidad-config  
  xmlns="http://myfaces.apache.org/trinidad/config">  
  <skin-family>MySkin</skin-family>  
</trinidad-config>
```

# trinidad-skins.xml Example

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<skins xmlns="http://myfaces.apache.org/trinidad/skin">
  <skin>
    <id>
      MySkin.desktop
    </id>
    <family>
      MySkin
    </family>
    <render-kit-id>
      org.apache.myfaces.trinidad.desktop
    </render-kit-id>
    <style-sheet-name>
      skins/blafplus-rich-extended.css
    </style-sheet-name>
    <extends>blafplus-rich.desktop
    </extends>
    <bundle-name>
      AdcsResourceBundle
    </bundle-name>
  </skin>
</skins>
```

# Creating a Custom Resource Bundle

- Replace default string labels of ADF Faces
  - Component labels
  - Validator message text
- Create message bundle class
  - Extending `java.util.ListResourceBundle`

```
public class AdcsResourceBundle extends ListResourceBundle {  
  
    public Object[][] getContents() { return contents; }  
  
    static final Object[][] contents =  
    {  
        {"af_dialog.OK", "Okay"},  
        {"af_panelWindow.CLOSE", "Close"},  
        {"af_document.SPLASH_SCREEN_MESSAGE", "Hello and Welcome"}};  
    }  
}
```

# Skinning Keys

- Skin Selector
- A skin key is used to style a component or components in an application, not a particular instance
- A skin key usually starts with af|, contains the name of the component, the piece of the component to skin, via a pseudo-element
  - ::label
  - ::content
  - ::read-only
- A skin key might look like styleclass, but ends with :alias
- skin keys can only be used in a Skinning CSS file
- Skin key cannot be used in a component's styleClass attribute

# Functionality Available to "Skinners"

- Skinning keys to skin pieces of a component, including icons
- Resource bundle keys to skin text of a component
- @platform {/skin definitions go here/}
  - Possible values are: windows, macos, linux, solaris, ppc
  - Does not work for icons
- @agent {/skin definitions go here/} - >
  - Possible values are: netscape, ie, mozilla, gecko, webkit , ice
  - Does not work for icons
- :lang or :locale
- :rtl pseudo-class to create a style or icon definition when the browser is in a right-to-left language
- :alias skinning keys

# Skin Custom Tree Icons

```
af|tree::expanded-icon
{
  content: url(/skins/mycompany/skin_images/expand.gif);
  width:16px;
  height:16px;
}

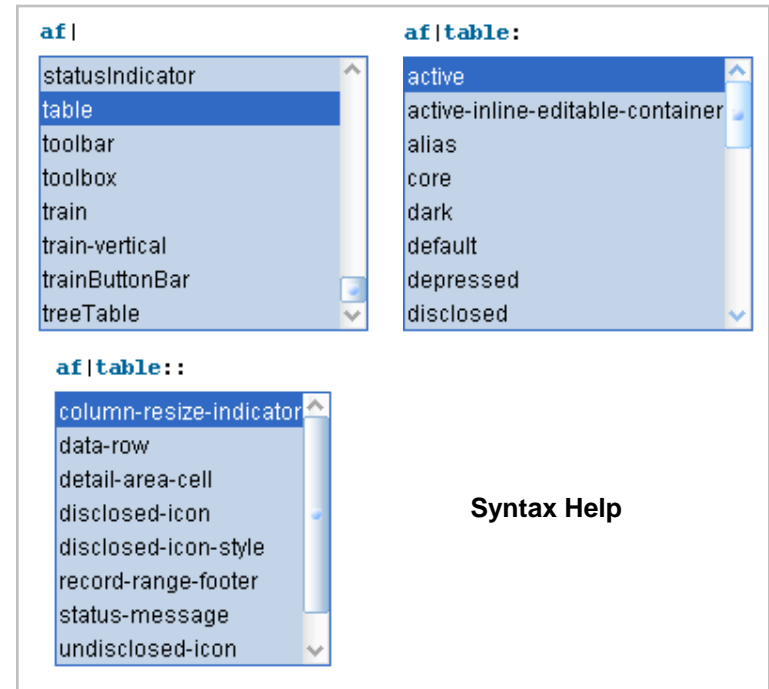
af|tree::collapsed-icon
{
  content: url(/skins/mycompany/collapse.gif);
  width:16px;
  height:16px;
}
```

# Agenda : Skinning

- About Skinning
- The Problem of directly added CSS
- How-to skin ADF Faces
- • Skinning Support in Oracle JDeveloper
- Debugging
- Demo

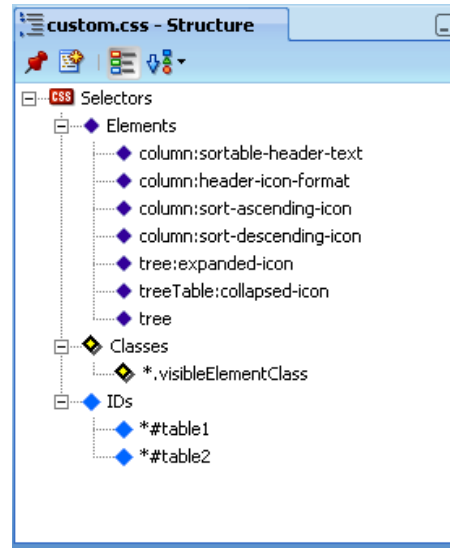
# Skin Development Support

- Code editing support
- ADF Faces Skin Extension
  - Tool > Preferences > CSS Editor
    - Check ADF Faces Extension
  - Syntax Help
  - Image selection
  - Code Completion
    - E.g. type "af|inputr" then press ctrl+Enter
  - Code Folding
    - Collapses CSS style definitions
    - Mouse-over code info



# Skin Development Support

- Structure Window Support
  - Classes
  - Elements
  - ID
  - Use to navigate and uncomment entries
- Error Margin
  - E.g. syntax error



# Agenda : Skinning

- About Skinning
- The Problem of directly added CSS
- How-to skin ADF Faces
- Skinning Support in Oracle JDeveloper
- • Debugging
- Demo

# Disable Content Compression

- web.xml

```
<context-param>  
  <param-name>  
    org.apache.myfaces.trinidad.DISABLE_CONTENT_COMPRESSION  
  </param-name>  
  <param-value>>true</param-value>  
</context-param>
```

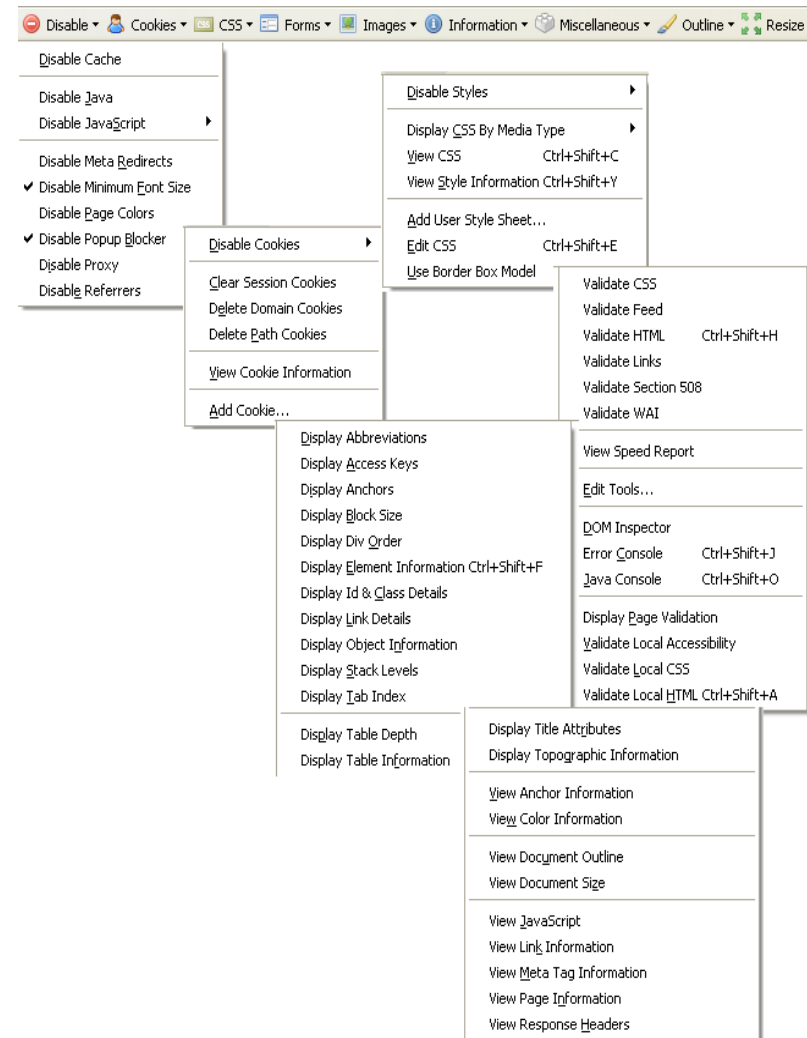
# Firebug



- Use within FireFox
- Inspect and edit HTML
- View and Visualize CSS
- Instantly change CSS definitions
- Exploring the DOM tree
- Execute JavaScript

# Web Developer Plugin

- Firefox and Mozilla
- Menu and a toolbar various web developer tools
- <http://chrispederick.com/work/web-developer/>



# Agenda : Skinning

- About Skinning
- The Problem of directly added CSS
- How-to skin ADF Faces
- Skinning Support in Oracle JDeveloper
- Debugging
- • Demo

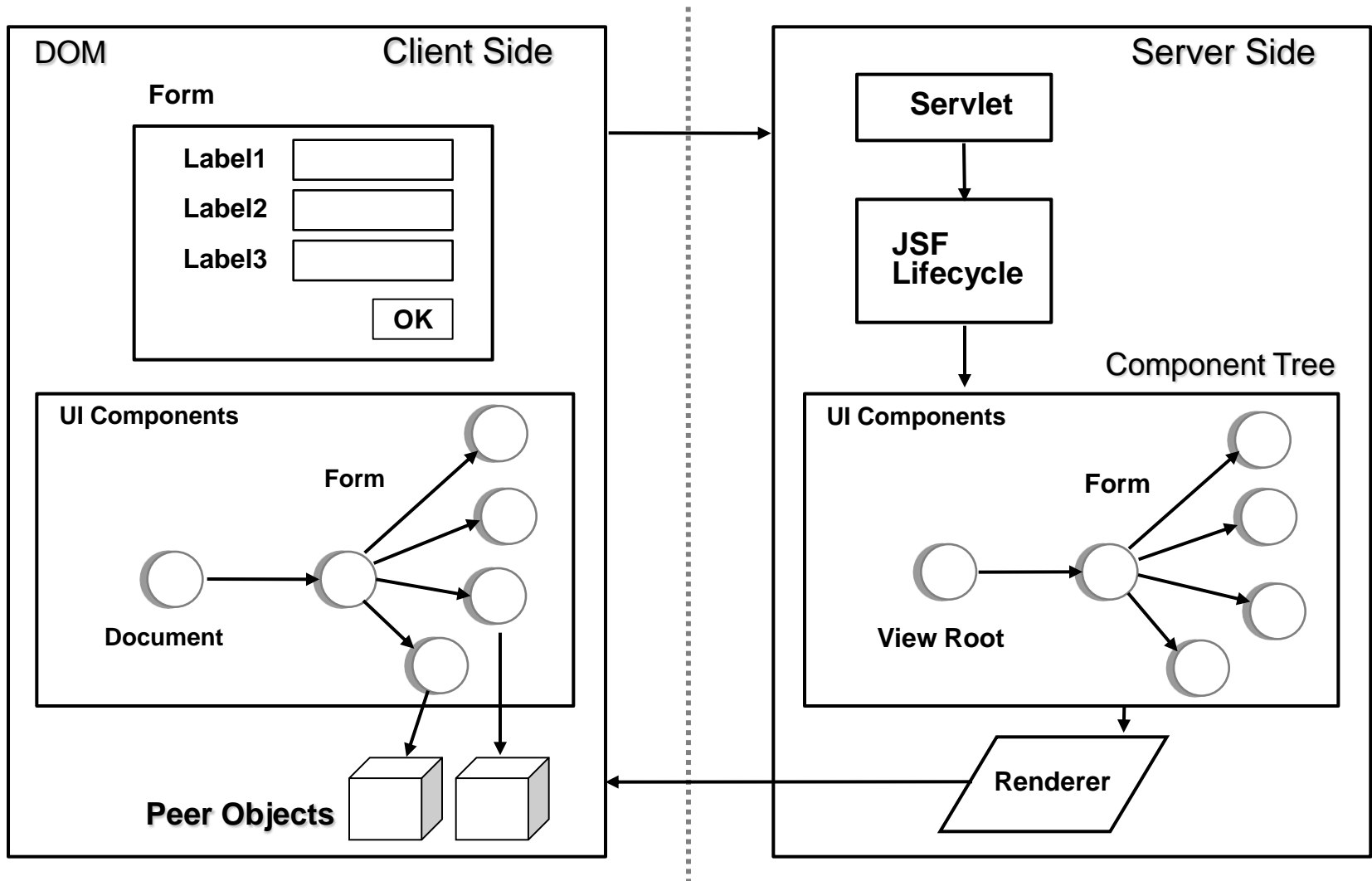
# Agenda : JavaScript

- • ADF JavaScript Architecture
- Listen for Component Events
- Invoking Server Side Logic
- Calling JavaScript from Java
- Debugging & Assertions
- Demo

# When to use JavaScript

- If a use case requires JavaScript
- If there is no native solution to a problem
- For client side integration

# ADF Faces Architecture



**AdfRichInputText**

Oracle Fusion Middleware JavaScript API Reference for Oracle ADF Faces  
11g Release 1 (11.1.1)  
E12046-03

All Classes

Packages

- [oracle.adf.view.js.agent](#)
- [oracle.adf.view.js.assert](#)
- [oracle.adf.view.js.base](#)
- [oracle.adf.view.js.component](#)
- [oracle.adf.view.js.component.fragment](#)
- [oracle.adf.view.js.component.rich](#)
- [oracle.adf.view.js.component.rich.data](#)
- [oracle.adf.view.js.component.rich.fragment](#)
- [oracle.adf.view.js.component.rich.input](#)
- [oracle.adf.view.js.component.rich.layout](#)
- [oracle.adf.view.js.component.rich.nav](#)
- [oracle.adf.view.js.component.rich.output](#)
- [oracle.adf.view.js.convert](#)
- [oracle.adf.view.js.data](#)
- [oracle.adf.view.js.data.transfer](#)
- [oracle.adf.view.js.dnd](#)

---

All Packages

- [AdfAbstractFloatingElement](#)
- [AdfActionEvent](#)
- [AdfAgent](#)
- [AdfAlertLogWriter](#)
- [AdfAssert](#)
- [AdfAttributeDragSource](#)
- [AdfAttributeDropTarget](#)
- [AdfAutoDismissalManager](#)
- [AdfAutoSubmitEvent](#)
- [AdfAutoSuggestBehavior](#)
- [AdfBaseEvent](#)
- [AdfBasicDropTarget](#)
- [AdfBootstrap](#)
- [AdfBufferedLogWriter](#)
- [AdfBusyStateEvent](#)
- [AdfCalendarActivity](#)
- [AdfCalendarActivityDurationChangeEvent](#)
- [AdfCalendarActivityEvent](#)
- [AdfCalendarDisplayChangeEvent](#)
- [AdfCalendarDragSource](#)
- [AdfCalendarDropTarget](#)
- [AdfCalendarEvent](#)
- [AdfCarouselDataFetchEvent](#)
- [AdfCarouselSpinEvent](#)
- [AdfCheckpoint](#)
- [AdfClientBehavior](#)
- [AdfClientFileLogWriter](#)
- [AdfCollection](#)

## ORACLE JavaScript API Reference for Oracle ADF Faces

Oracle Fusion Middleware JavaScript API Reference for Oracle ADF Faces  
11g Release 1 (11.1.1)  
E12046-03

[Overview](#)
[Package](#)
[Class](#)
[Tree](#)
[Deprecated](#)
[Index](#)
[Help](#)

[PREV](#)
[NEXT](#)
[FRAMES](#)
[NO FRAMES](#)

---

SUMMARY: [FIELD](#) | [CONSTR](#) | [METHOD](#)    DETAIL: [FIELD](#) | [CONSTR](#) | [METHOD](#)

---

### oracle.adf.view.js.component.rich.input Class AdfRichInputText

[org.ecmascript.object.Object](#)

```

|--org.apache.myjs.trinidad.component.AdfUIValue
|
|--org.apache.myjs.trinidad.component.AdfUIEditableValue
|
|   |--org.apache.myjs.trinidad.component.AdfUIInput
|   |
|   |--oracle.adf.view.js.component.rich.input.AdfRichInputText

```

#### Method Summary

public <a href="#">String</a>	<a href="#">getAccessKey()</a> Get function for attribute for 'accessKey'.
public <a href="#">Boolean</a>	<a href="#">getAutoSubmit()</a> Get function for attribute for 'autoSubmit'.
public <a href="#">Boolean</a>	<a href="#">getAutoTab()</a> Get function for attribute for 'autoTab'.
public <a href="#">Boolean</a>	<a href="#">getChanged()</a> Get function for attribute for 'changed'.
public <a href="#">String</a>	<a href="#">getChangedDesc()</a> Get function for attribute for 'changedDesc'.
public <a href="#">Number</a>	<a href="#">getColumns()</a> Get function for attribute for 'columns'.
public <a href="#">String</a>	<a href="#">getContentStyle()</a> Get function for attribute for 'contentStyle'.
public <a href="#">Boolean</a>	<a href="#">getDisabled()</a> Get function for attribute for 'disabled'.
public <a href="#">String</a>	<a href="#">getHelpTopicId()</a> Get function for attribute for 'helpTopicId'.
public <a href="#">String</a>	<a href="#">getInlineStyle()</a> Get function for attribute for 'inlineStyle'.
public <a href="#">String</a>	<a href="#">getLabel()</a> ...

# Agenda : JavaScript

- ADF JavaScript Architecture
- • Listen for Component Events
- Invoking Server Side Logic
- Calling JavaScript from Java
- Debugging & Assertions
- Demo

# af:resource

- Adds CSS or JavaScript resource to HTML header
- Child component of af:document
- Processed on the initial creation of the document
- If source attribute is not defined, content of the tag is used

```
<af:document>
  <af:resource type="javascript" source="/customJsCode.js"/>
  ...
</af:document>
```

```
<af:document>
  <af:resource type="javascript">
    function customJsFunction(){ ... }
  </af:resource>
  ...
</af:document>
```

# How-to find ADF Faces Components on a Page

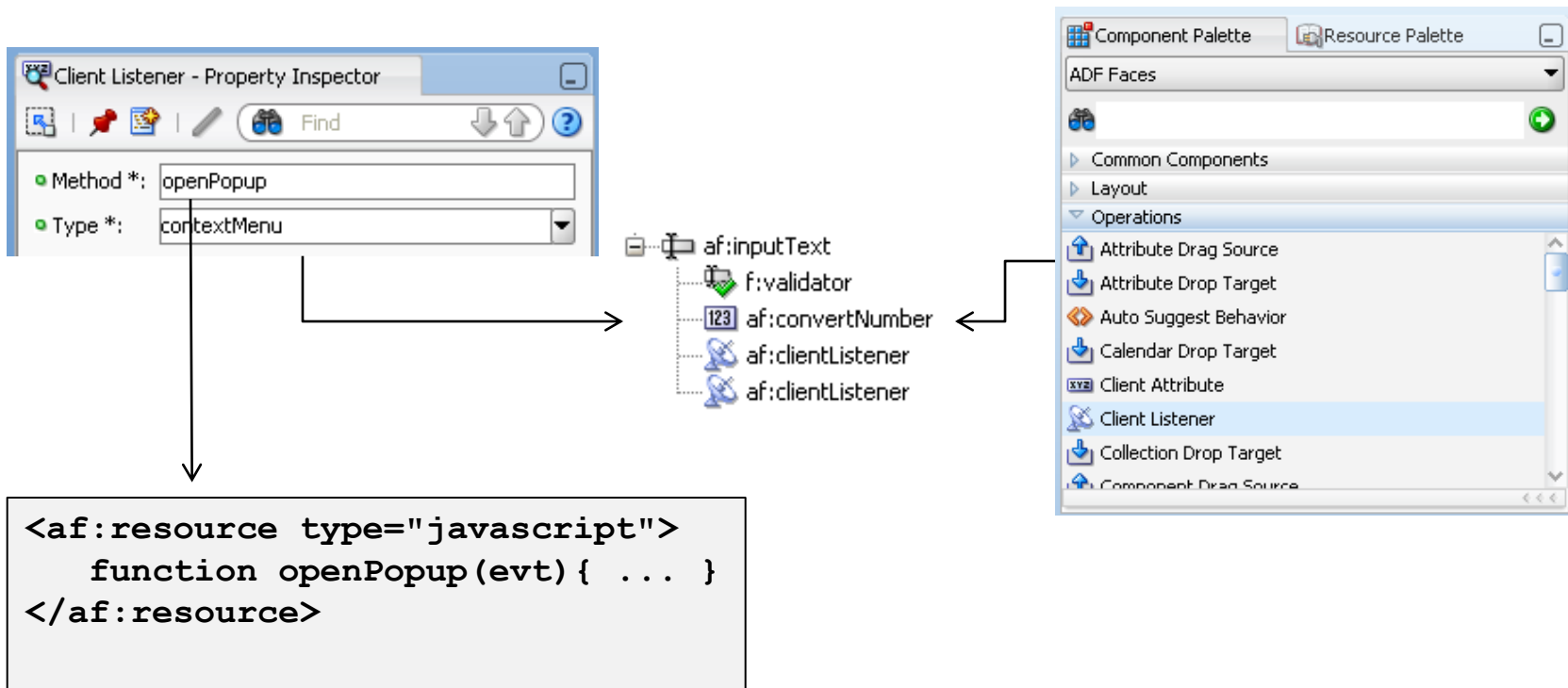
- Search from AdfPage.PAGE
  - findComponentByAbsoluteId
    - Searches by the component ID
  - findComponentByAbsoluteLocator
    - Searches in component that stamps its children
- Relative Search
  - findComponent
  - Searches the component clientId
  - ClientId is renderer implementation specific
- Be aware of naming containers
- To ensure client component exists
  - Set clientComponent property to true, ...
  - Apply af:clientListener to component

# Component Events

- Component events wrap component and DOM events
- Component event object names start with "Adf", followed by the event name and end with "Event"
- Allow developers to listen and respond to component changes
  - value change, disclosure, query, focus, popup ...
- Events provide information about
  - Component the event is invoked on
  - The event object
- Listened for with `af:clientListener`
- Can be cancelled

# af:clientListener

- Method - defines the JavaScript function to call
- Type - defines the component event to listen for



# Agenda : JavaScript

- ADF JavaScript Architecture
- Listen for Component Events
- • Invoking Server Side Logic
- Calling JavaScript from Java
- Debugging & Assertions
- Demo

# Invoking Java on the Server

- Common requirement in Ajax
- Implemented through `af:serverListener` in ADF Faces
- Tag has two attributes
  - "type" attribute defines a named handle to call from JavaScript
  - "method" attribute references managed bean method using EL
- Messages can be sent to server using key/value pairs
  - {arg1:value1 arg2:value2 ...}
- Requests can be set to "immediate"

# af:serverListener

1

```
<af:inputText id="it1" label="...">
  <af:clientListener method="handleKeyUp" type="keyUp"/>
  <af:serverListener type="MyCustomServerEvent"
    method="#{mybean.handleServerEvent}"/>
</af:inputText>
```

2

```
<af:resource type="javascript">
  function handleKeyUp (evt) {
    var inputTextComponent = evt.getSource();
    AdfCustomEvent.queue(inputTextComponent,
      " MyCustomServerEvent ",
      {fvalue:component.getSubmittedValue()}
      ,false);

    evt.cancel();
  }
</af:resource>
```

3

```
public void handleServerEvent(ClientEvent ce){
  String message = (String) ce.getParameters().get("fvalue");
  ...
}
```

# Agenda : JavaScript

- ADF JavaScript Architecture
- Listen for Component Events
- Invoking Server Side Logic
- • Calling JavaScript from Java
- Debugging & Assertions
- Demo

# Invoking JavaScript from Java

- ExtendedRenderkitService
- Apache Trinidad utility class

# Invoking client side JavaScript from Java

- Calling JavaScript from Java

```
private void writeJavaScriptToClient(String script) {
    FacesContext fctx = FacesContext.getCurrentInstance();
    ExtendedRenderKitService erks = null;
    erks = Service.getRenderKitService(fctx,
                                       ExtendedRenderKitService.class);
    erks.addScript(fctx, script);
}
```

- Example

```
...
StringBuilder script = new StringBuilder();
script.append(
    "var popup = AdfPage.PAGE.findComponentByAbsoluteId('p1');");
script.append("if(popup != null){");
script.append("popup.show();");
script.append("}");
writeJavaScriptToClient(script.toString());
```

# Agenda : JavaScript

- ADF JavaScript Architecture
- Listen for Component Events
- Invoking Server Side Logic
- Calling JavaScript from Java
- • Debugging & Assertions
- Demo

# Debug JavaScript

- Disable JavaScript obfuscation in web.xml

```
<context-param>  
  <param-name>  
    org.apache.myfaces.trinidad.DEBUG_JAVASCRIPT  
  </param-name>  
  <param-value>>false</param-value>  
</context-param>
```

# Client side logging

- Web.xml parameter: oracle.adf.view.rich.LOGGER\_LEVEL
  - SEVERE      – WARNING
  - INFO        – CONFIG
  - FINE        – FINER
  - FINEST     – ALL

```
<context-param>
  <param-name>oracle.adf.view.rich.LOGGER_LEVEL</param-name>
  <param-value>ALL</param-value>
</context-param>
```

- Use in custom JavaScript

```
AdfLogger.LOGGER.logMessage(AdfLogger.<log level>,
                             'your message here');
```

# Assertions

- JavaScript is not a typed language
  - Functions must ensure the correct argument types are passed in
- ADF Faces provides JavaScript assert functions
  - Used by framework
  - Can be used in custom JS code
  - Throws `org.ecmascript.object.error.Error`
  - Error displays in browser
- Should be disabled in production environments
  - Disabled by default

# Assertions

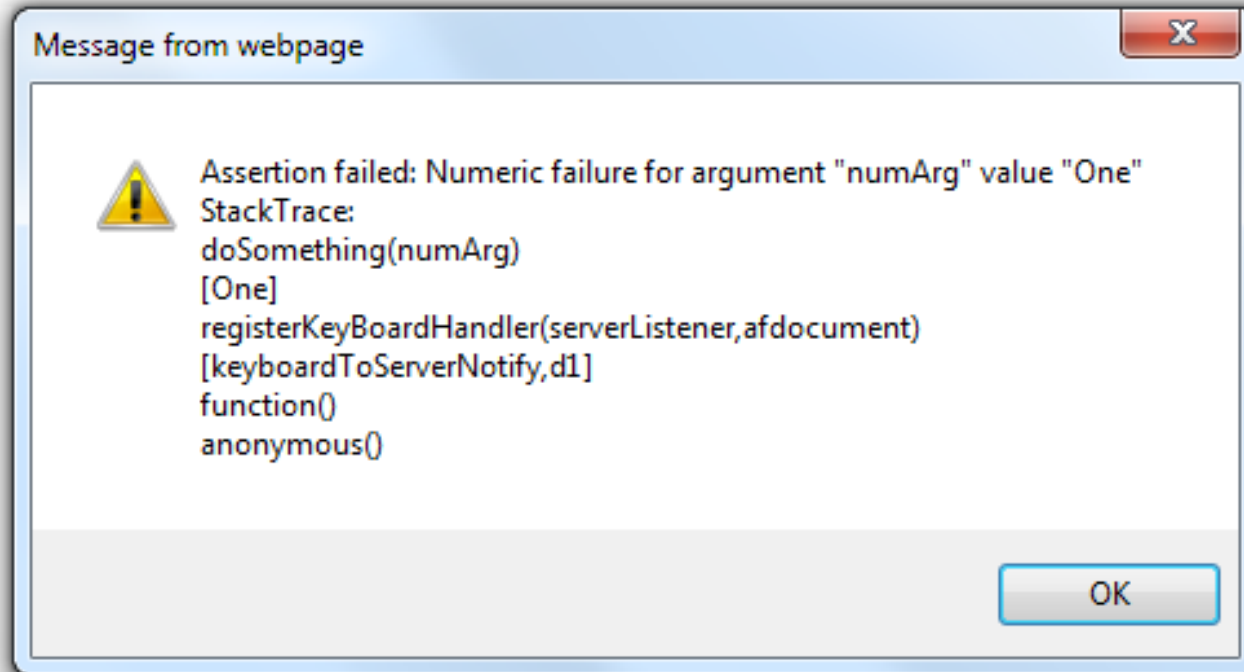
- web.xml context parameter

```
<context-param>
  <param-name>
    oracle.adf.view.rich.ASSERT_ENABLED
  </param-name>
  <param-value>true</param-value>
</context-param>
```

- How-to use in JavaScript function

```
function doSomething(numArg) {
  AdfAssert.assertNumeric
    (numArg, "Numeric failure for argument \"numArg\" value
      \" "+numArg+"\"");
  ... function code to follow here ...
}
```

# Assertions



# Agenda : JavaScript

- ADF JavaScript Architecture
- Listen for Component Events
- Invoking Server Side Logic
- Calling JavaScript from Java
- Debugging & Assertions
- • Demo

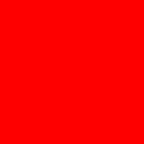
# For more information

- <http://www.oracle.com/technetwork/jdev>
- Downloads on Oracle Technology Network (OTN)
- Product Documentation
- Data Sheets and Whitepapers
- Blogs
- OTN Discussion Forums
- Books

# Learn More at ODTUG K-Scope 2011

- <http://kscope11.com/>
- Special Fusion Middleware Track
- 50 dedicated sessions and hands on labs
- Presented by Oracle ACEs, developers and product managers
- ADF, WebCenter, SOA Suite and more





The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

**ORACLE®**