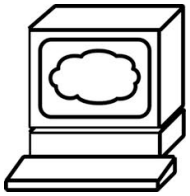
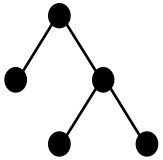


Architecture des Systèmes d'Information

Java Server Faces
JSF 2.0
introduction



Traduction en cours

JSF

JSF is "Swing for server-side applications".

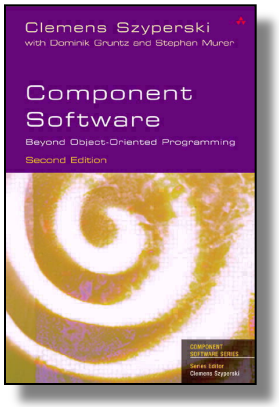
Server Side HMI Component Model

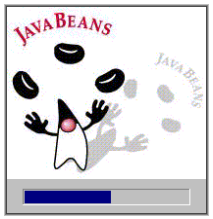
=

Client Side HMI Component Model

JEE (specifications)

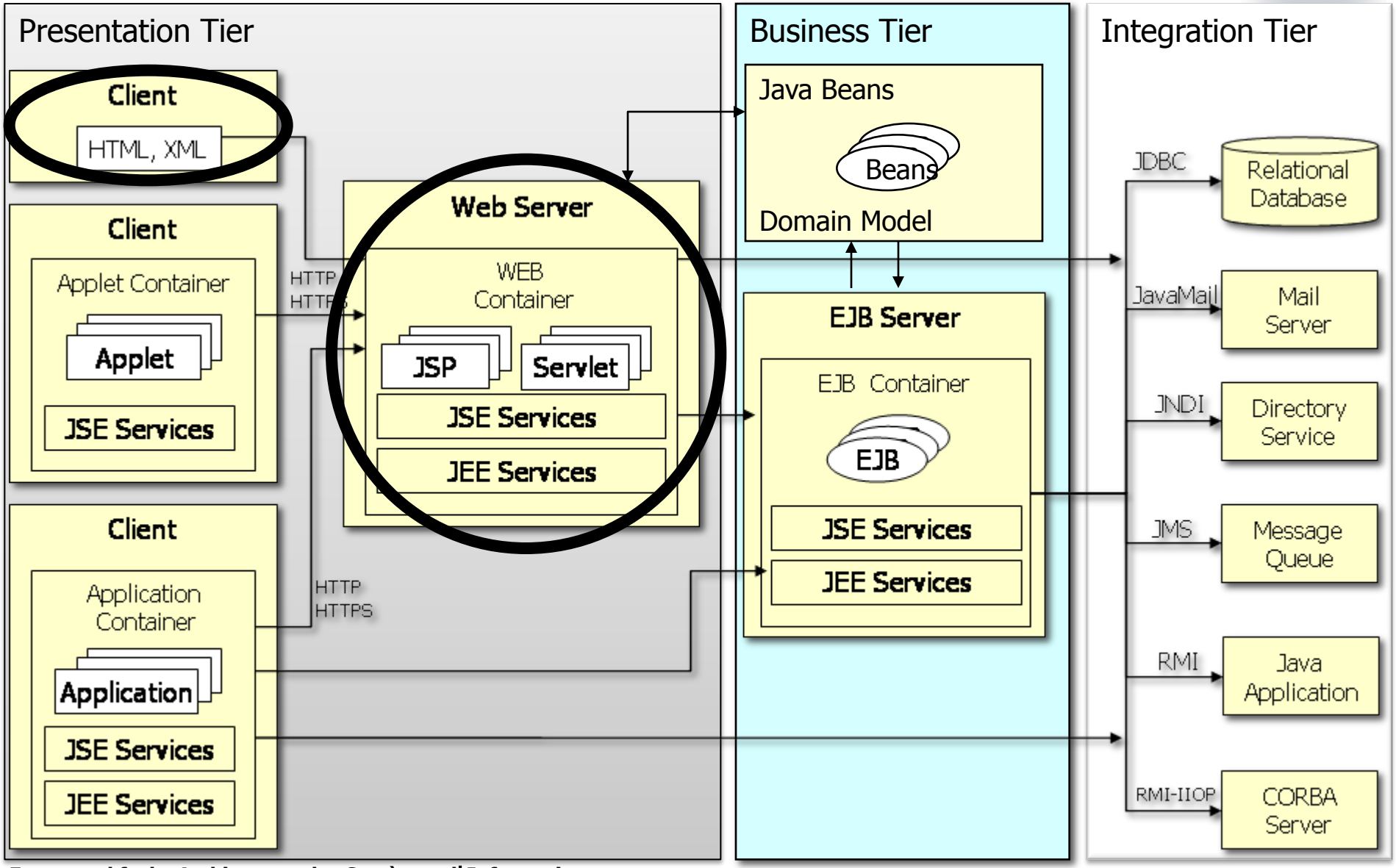
- Composants (Components)
  - Applet
  - Servlet
  - Java Server Page (JSP)
  - JSF (Java Server Faces)
  - Java Beans
  - Enterprise Java Beans (EJB)
  - Java Persistence API (JPA)





JEE and JSF

Server + Client

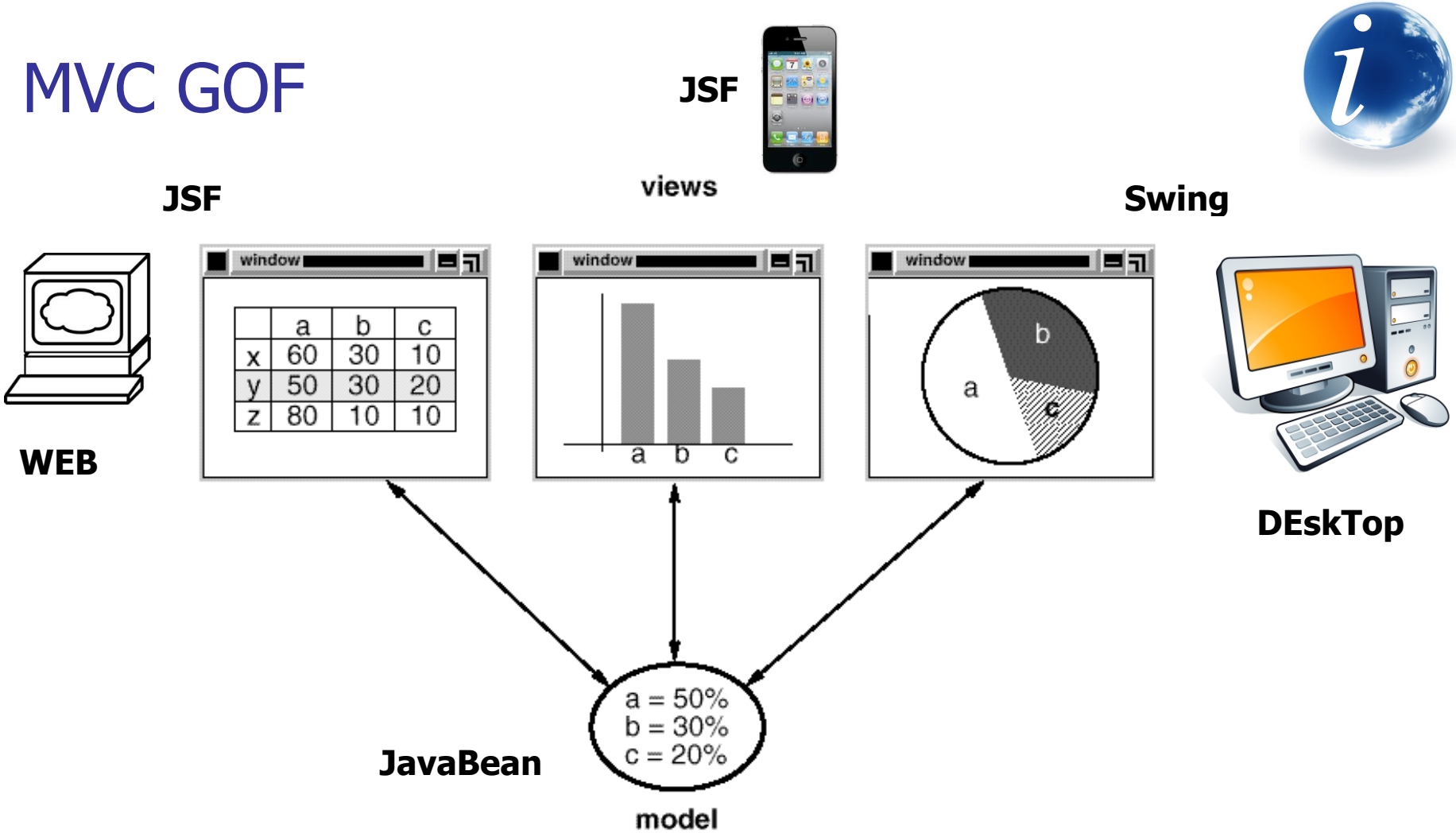


Java Server Faces Presentation Layer

- JSF is "Swing for server-side applications".
- JSF has three parts :
  - Data management using Java :
    - Java Bean : Managed Bean
  - A set of prefabricated UI components that are translated into standard HTML tags so that they can be used with any browser
  - A component model that enables third-party developers to supply additional components

http://www.d.umn.edu/~tcolburn/cs4531/slides/jsf2/ch1/ch1.xhtml

MVC GOF



MVC is really old and was first described in 1979 by Trygve Reenskaug, then working on Smalltalk at Xerox PARC.

Web Application Framework (WAF)

- Separation of content
- Components
- Events
- Validators
- Back-end-data integration
- Drag-and-Drop Visual Components

JSF versions

- JSF 2.1 : 2010-10-22
- JSF 2.0 : 2009-06-28
- JSF 1.2 : 2006-05-11
- JSF 1.1 : 2004-05-27
- JSF 1.0 : 2004-03-11

Facelet ui:

- JSF is web framework and Facelet is a view template for JSF.
- Facelet : Html include
- Allow to create a custom component without writing java code which extend Uicomponent.

http://www.oracle.com/technetwork/articles/java/facelets-454361.html

Server-side tags

- Conditional processing
- Iteration
- Output
- Variable setting
- Functions
- …

TAGs programing

- Server-side tags
- Separation of concerns
- Separation of content
- Tag-based Web application development
- Custom Tags
- Tags library
- Java implementation
- XML implementation

JSTL : Java Standard Tag Library

- JSP 1.1
  - JSP extension based on Custom TAG.
  - TAG are similar to JSP action, but replace JSP: prefixe with user define prefix.
- Common Extension are gathered in JSP Standard TAG Library.
- TAGs are interpreted by the servlet container.
- TAGs are replacing Java Scriplets.

JAVASERVER PAGES™
DYNAMICALLY GENERATED WEB CONTENT

Reminder JSP Actions

- <jsp:include/>
- <jsp:forward/>
- <jsp:plugin/>
- <jsp:usebean/>
- <jsp:setProperty/>
- <jsp:getProperty/>

Tag Library	URI	Prefix	Example	Contents
JavaServer Faces Facelets Tag Library	http://java.sun.com/jsf/facelets	ui:	ui:component ui:insert	Tags for templating
JavaServer Faces HTML Tag Library	http://java.sun.com/jsf/html	h:	h:head h:body h:outputText h:inputText	JavaServer Faces component tags for all UIComponent objects
JavaServer Faces Core Tag Library	http://java.sun.com/jsf/core	f:	f:actionListener f:attribute	Tags for JavaServer Faces custom actions that are independent of any particular render kit
JSTL Core Tag Library	http://java.sun.com/jsp/jstl/core	c:	c:forEach c:catch	JSTL 1.2 Core Tags
JSTL Functions Tag Library	http://java.sun.com/jsp/jstl/functions	fn:	fn:toUpperCase fn:toLowerCase	JSTL 1.2 Functions Tags

Tag Libraries

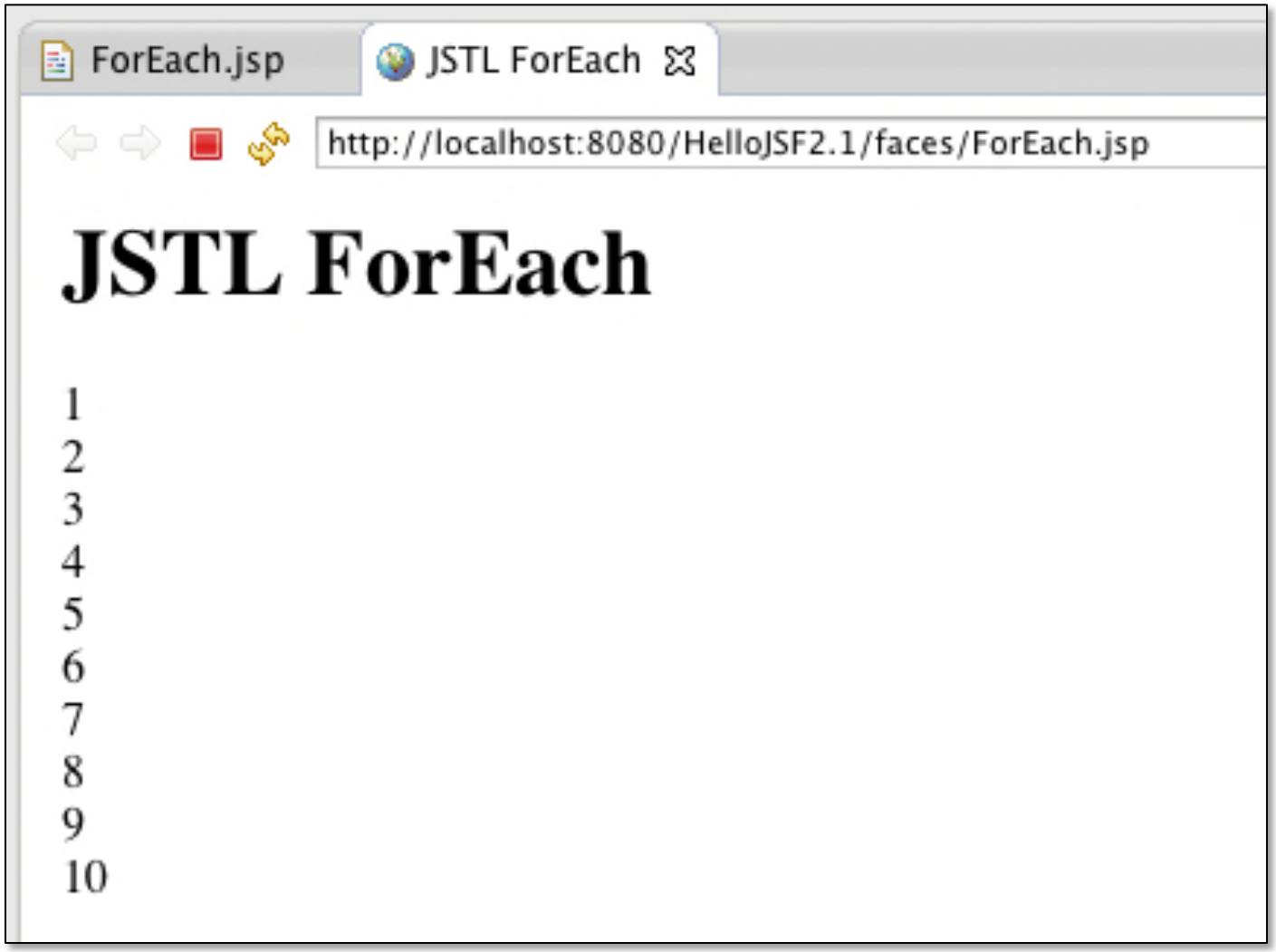
- f :
  - The core JavaServer Faces custom actions that are independent of any particular RenderKit.
- h :
  - This tag library contains JavaServer Faces component tags for all UIComponent
  - HTML RenderKit Renderer combinations defined in the JavaServer Faces Specification.
- ui :
  - The tags in this library add templating—a powerful view composition technique—to JSF.

TAGs programing : ForEach

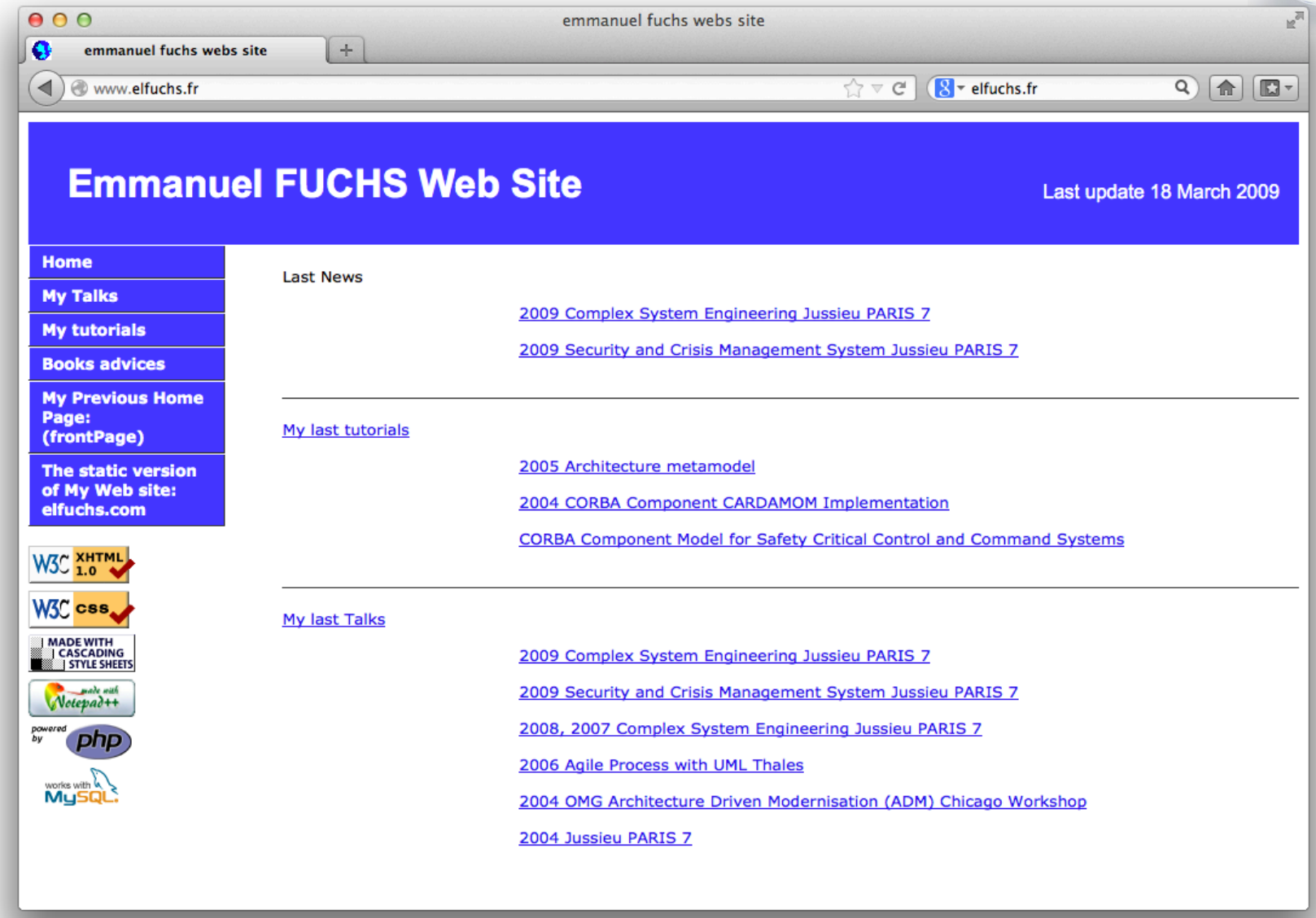
ForEach.jsp | JSTL ForEach

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<html>
<head>
<title>JSTL ForEach</title>
</head>
<body>
    <h1>JSTL ForEach</h1>

    <c:forEach var="i" begin="1" end="10" step="1">
        <c:out value="${i}" />
        <br />
    </c:forEach>

</body>
</html>
```

TAGs programing



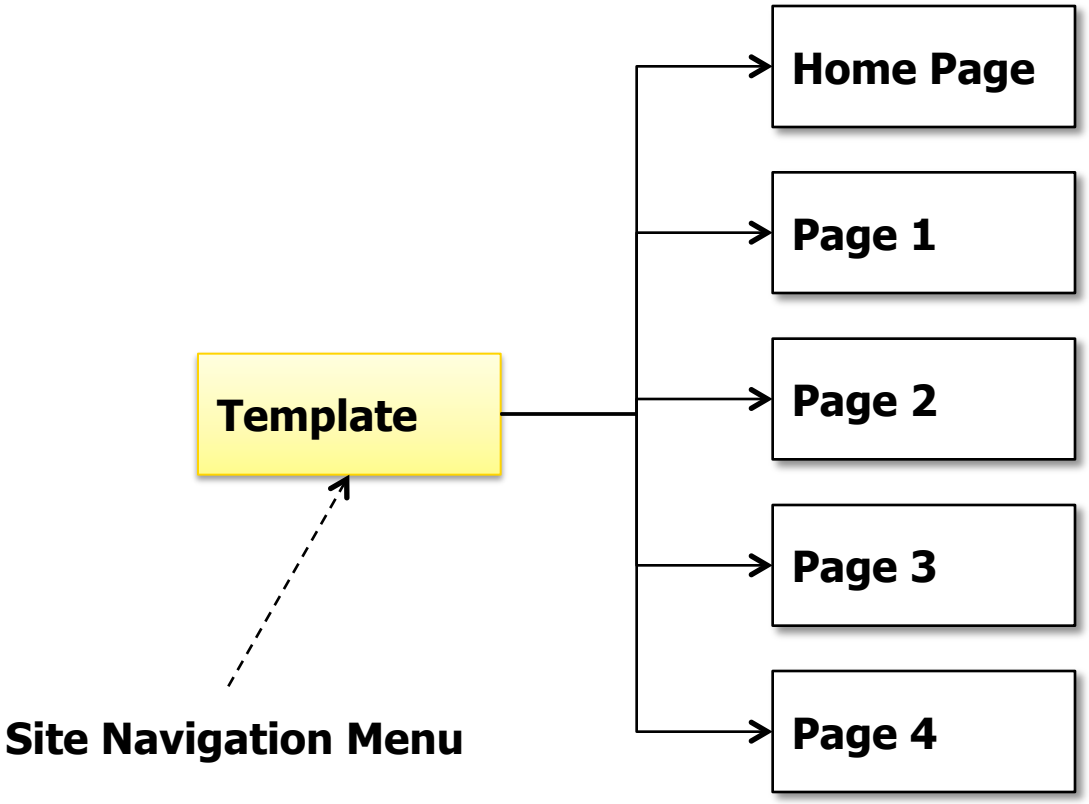
Html template : ?



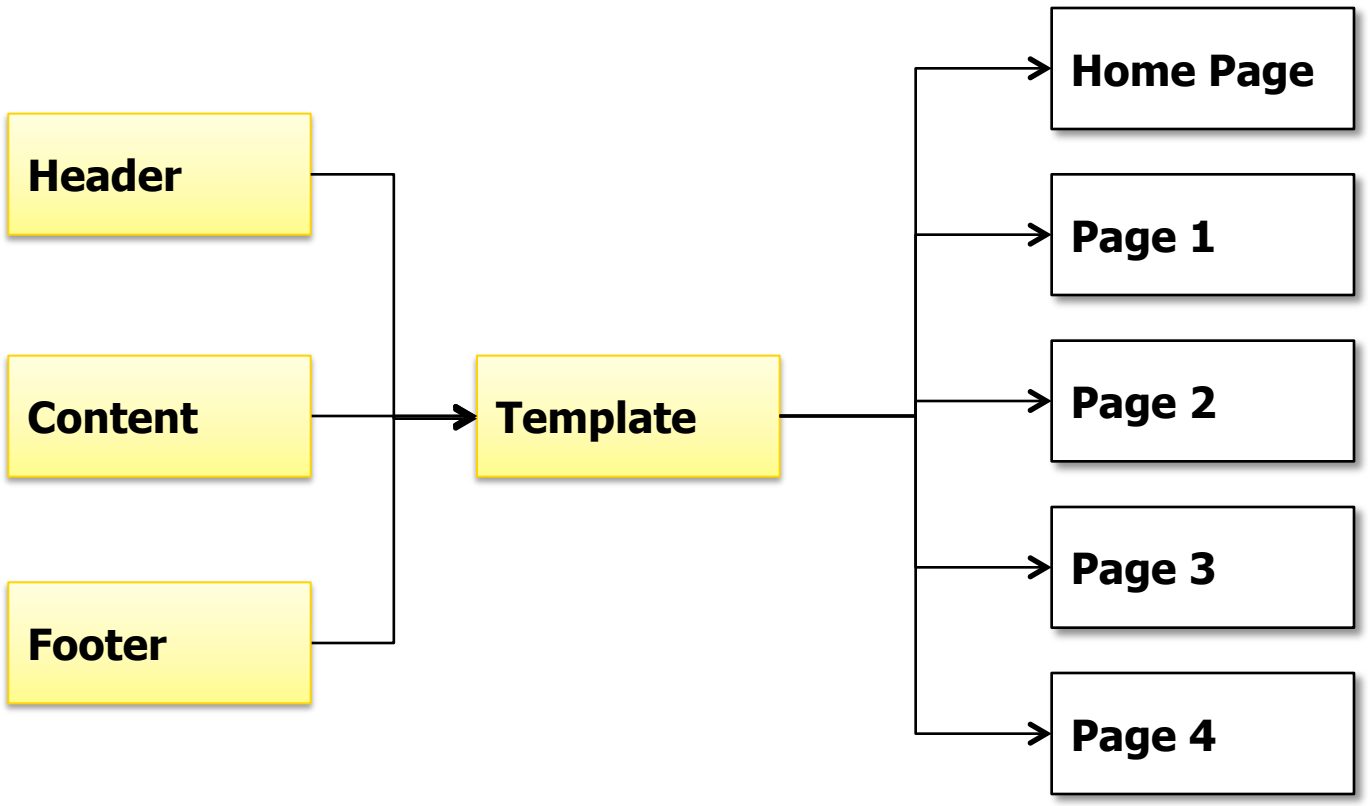
Facelet Tag

- h:outputStylesheet
  - to include a CSS file stylesheet applicable to the whole site pages.
- ui:insert
  - defines content to be replace by the file that embarks the template.
- ui:define
  - Defines content to be insert into template.
- ui:include
  - includes content from another XHTML page.
- ui:composition
  - children of this tag defines the template layout.

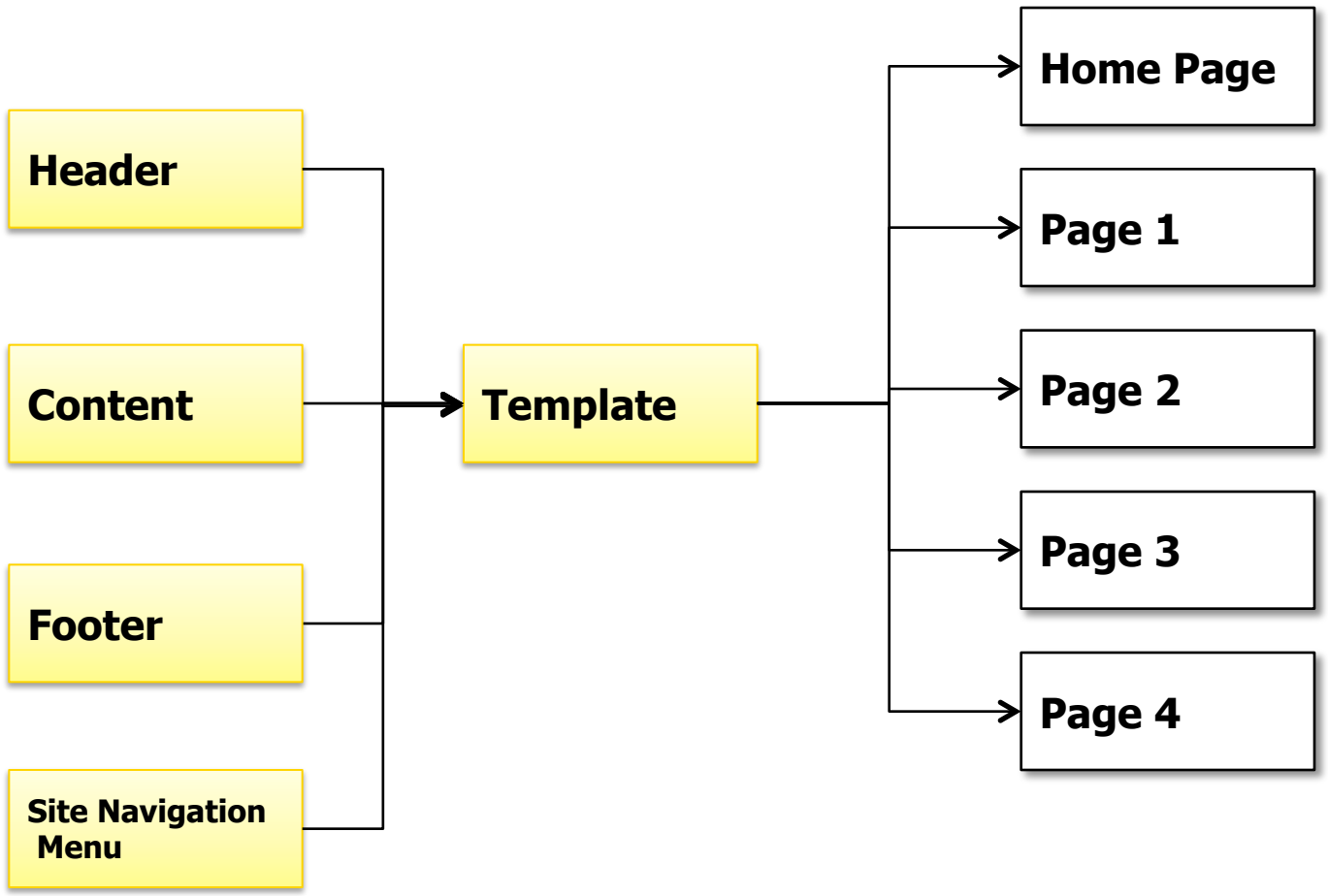
JSF Template



JSF Template



JSF Template Modularity



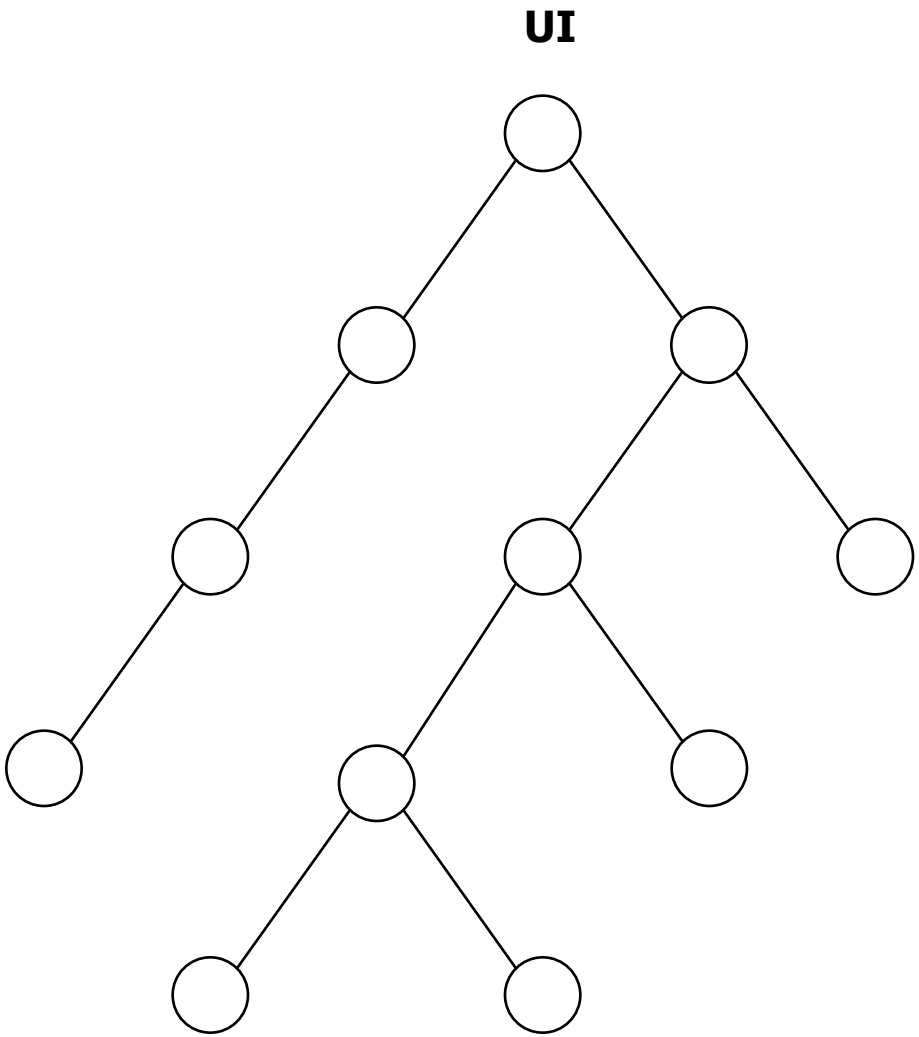
Origin

- Products
  - ColdFusion
  - Struts
  - ASP
- JSP
  - JSP standard Action
  - Custom Tags (Struts Taglibs)
  - JSTL (JSTL : Java Standard Tag Library)
    - Expression language (EL)
- JSF : faces
  - JSP independent

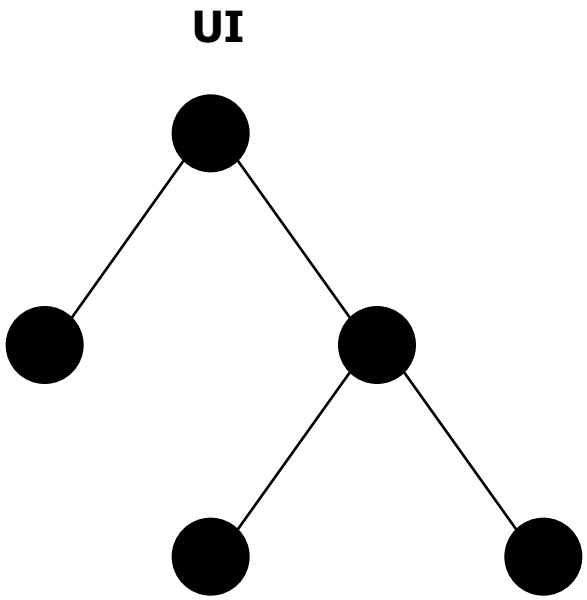
UI components tree

- A JSF page is represented by a tree of UI components, called a view.
- Same principle as a web browser DOM in client side.
- UI client side are statefull objects on the server side

User Interface (UI) tree



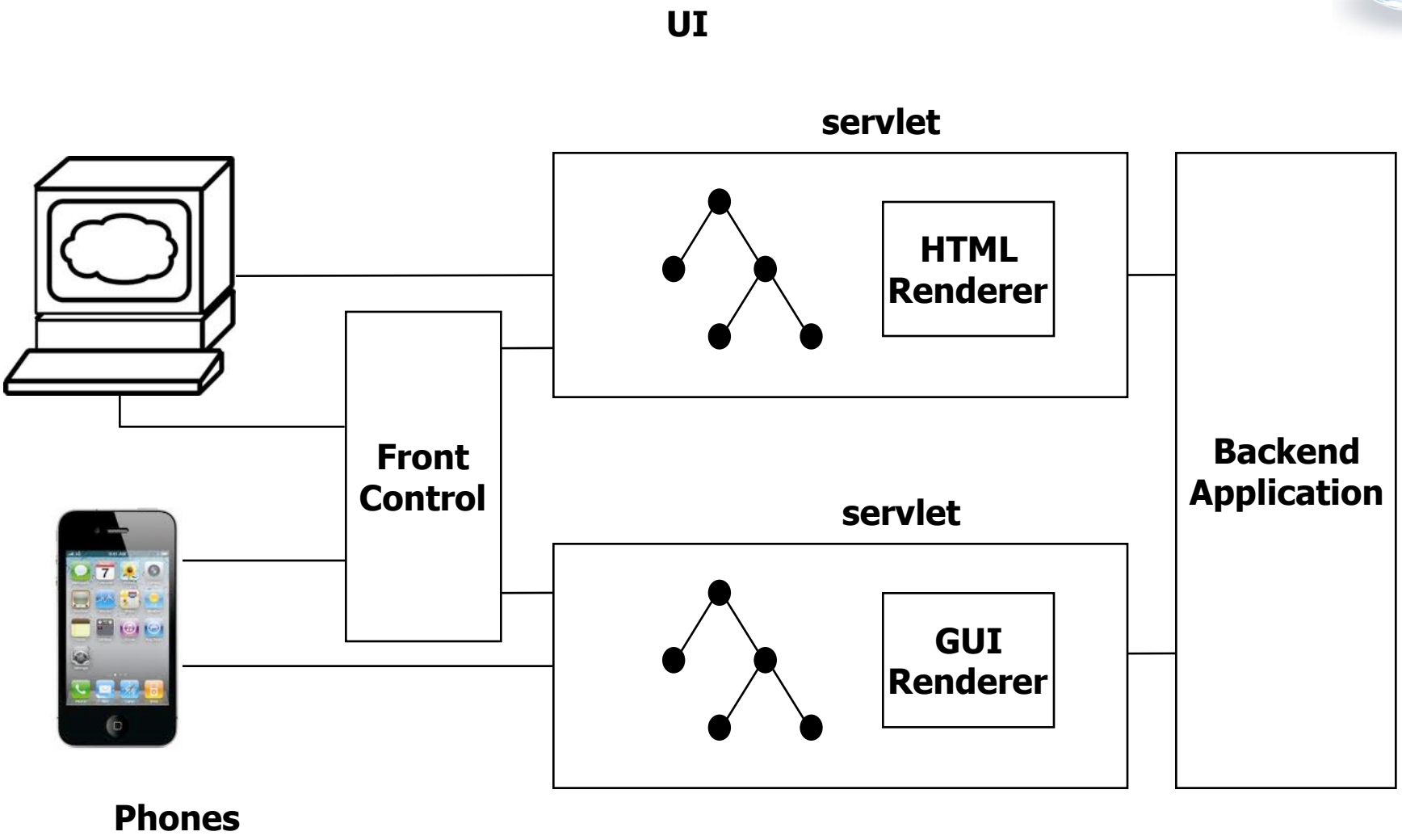
Uuser Interface (UI) tree



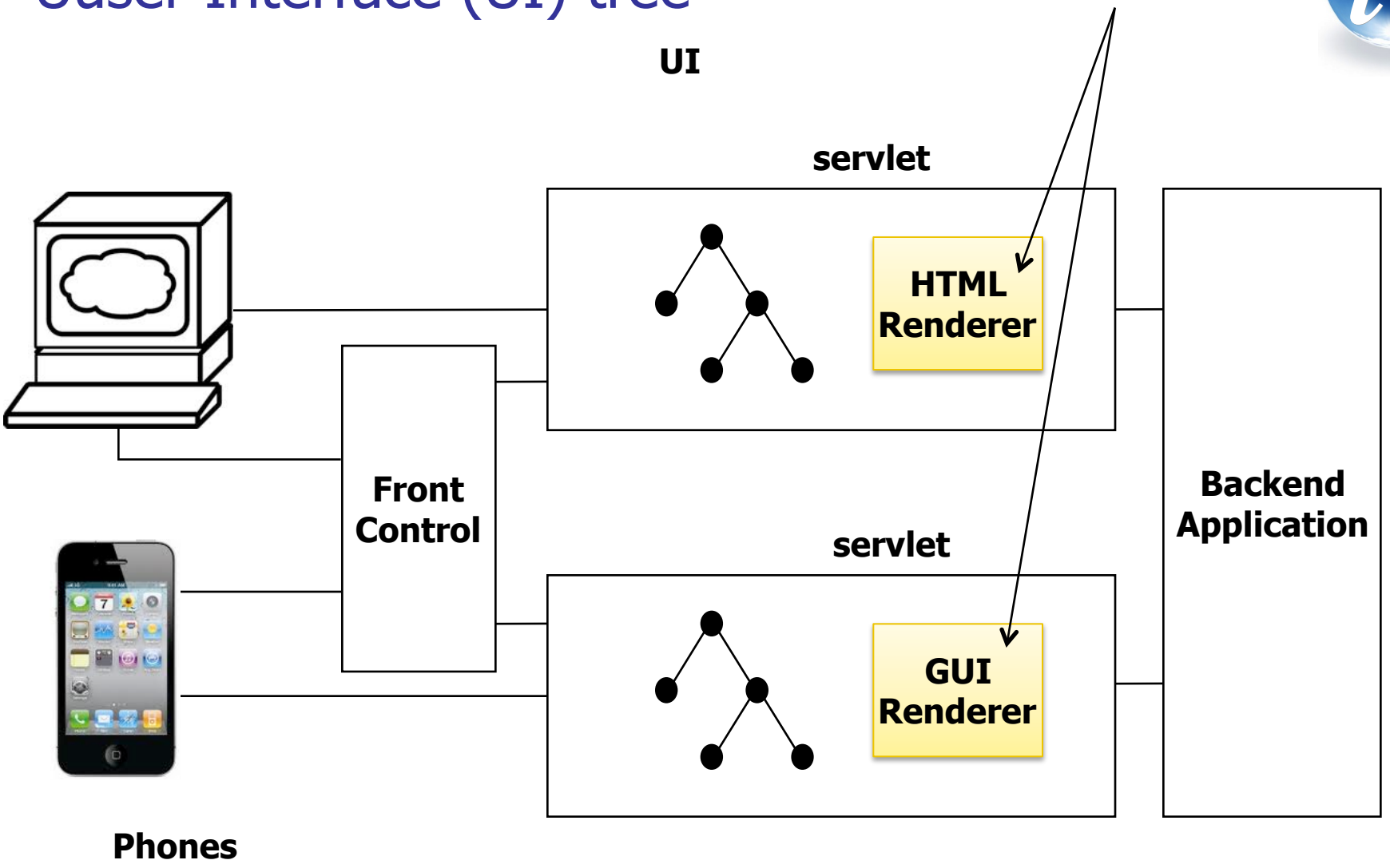
Abstract Component Tree : ACT

- Client device independence :
  - PDA, Desktop, Phone
  - Independent of the target implementation
    - HTML
    - …
- To be leveraged by drag-and-drop graphical interface design tools.
- Standard UI framework

Uuser Interface (UI) tree



Uuser Interface (UI) tree



JSF LifeCycle

- 1 Restore View
- 2 Apply requests Process events
- 3 Process Validations
- 4 Update model values Process events
- 5 Invoke applications Process events
- 6 Render response

javax.faces
com.sun.faces.lifecycle

JSF Components

- In JSF, a component is a group of interacting classes that together provide a reusable piece of web-based user interface code.
- A component is made up of three classes that work closely together :
  - The Renderer
  - the UIComponent
  - Dynamic Action

http://today.java.net/pub/a/today/2004/07/16/jsfcustom.html

Renderer

- Creates the client-side representation of the component.
- Manages input from the client and transforms it into something the component can understand.
- Renderer implementation
  - HMTL renderer implementation will generate HTML
  - XUL renderer implementation will generate HTML.

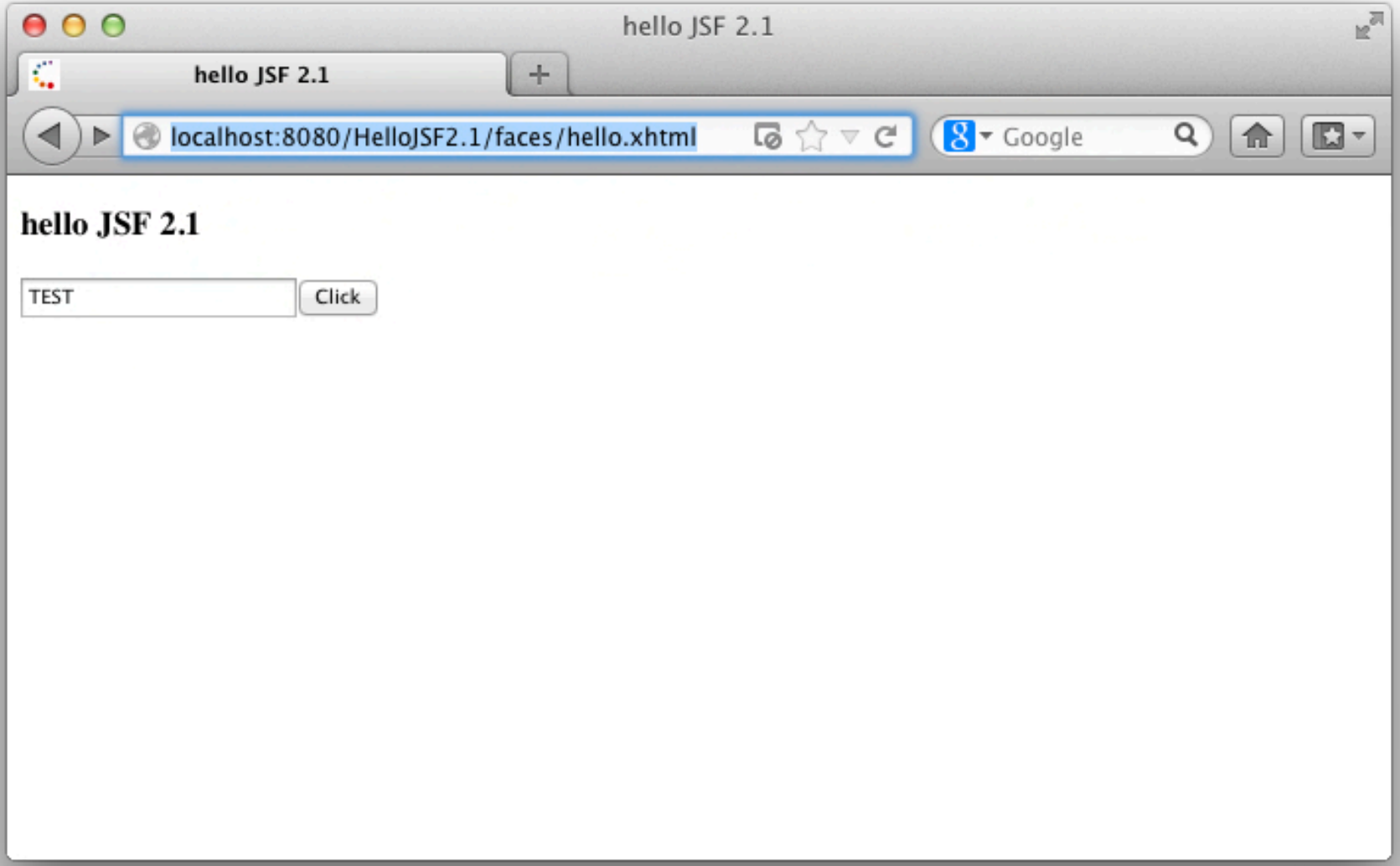
UIComponent

- The component asiociated class is responsible for the data and behavior of the component on the server side.

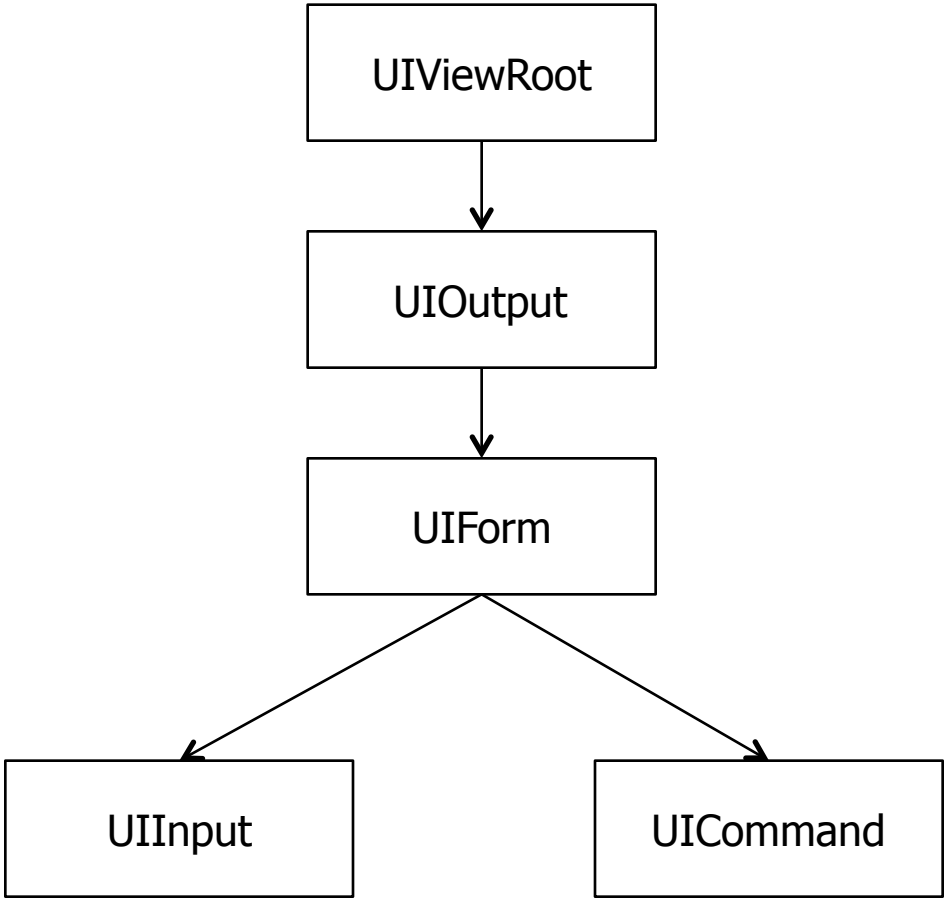
Component associated Action

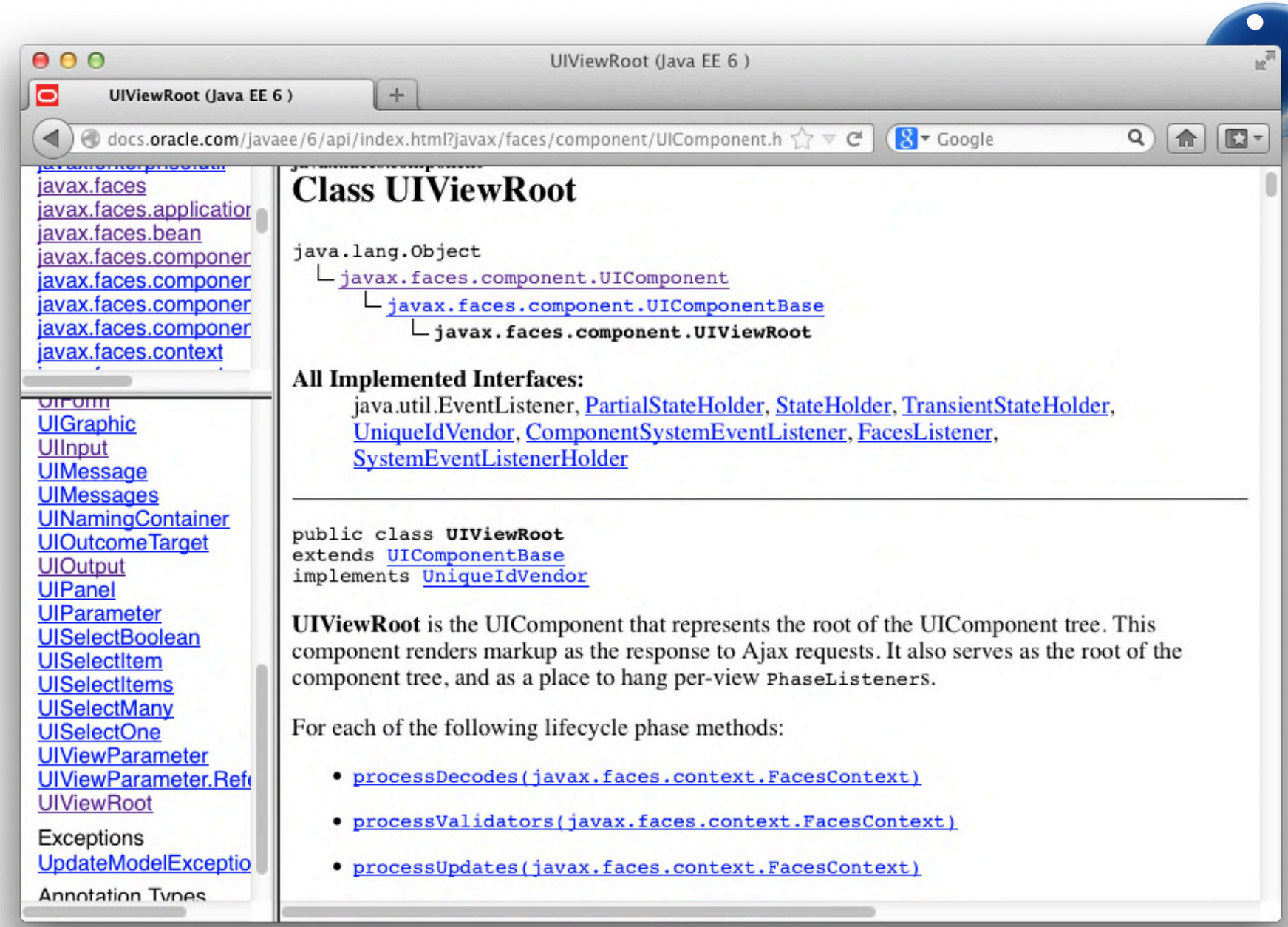
- Association between component and action.
- Exemple :
  - h:commandButton will render a button and
  - h:commandLink will render an HTML link.

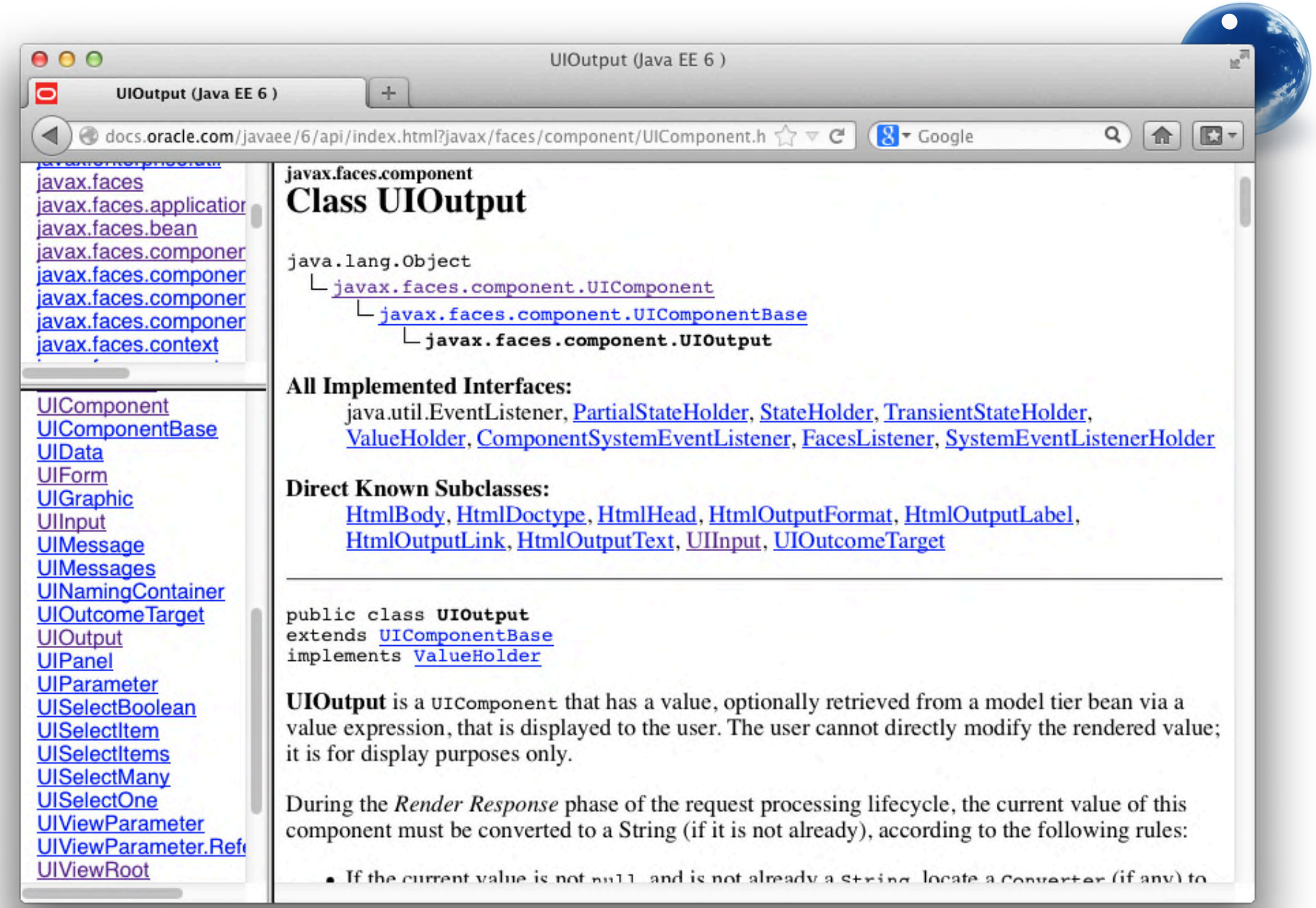
JSF UI component tree exemple

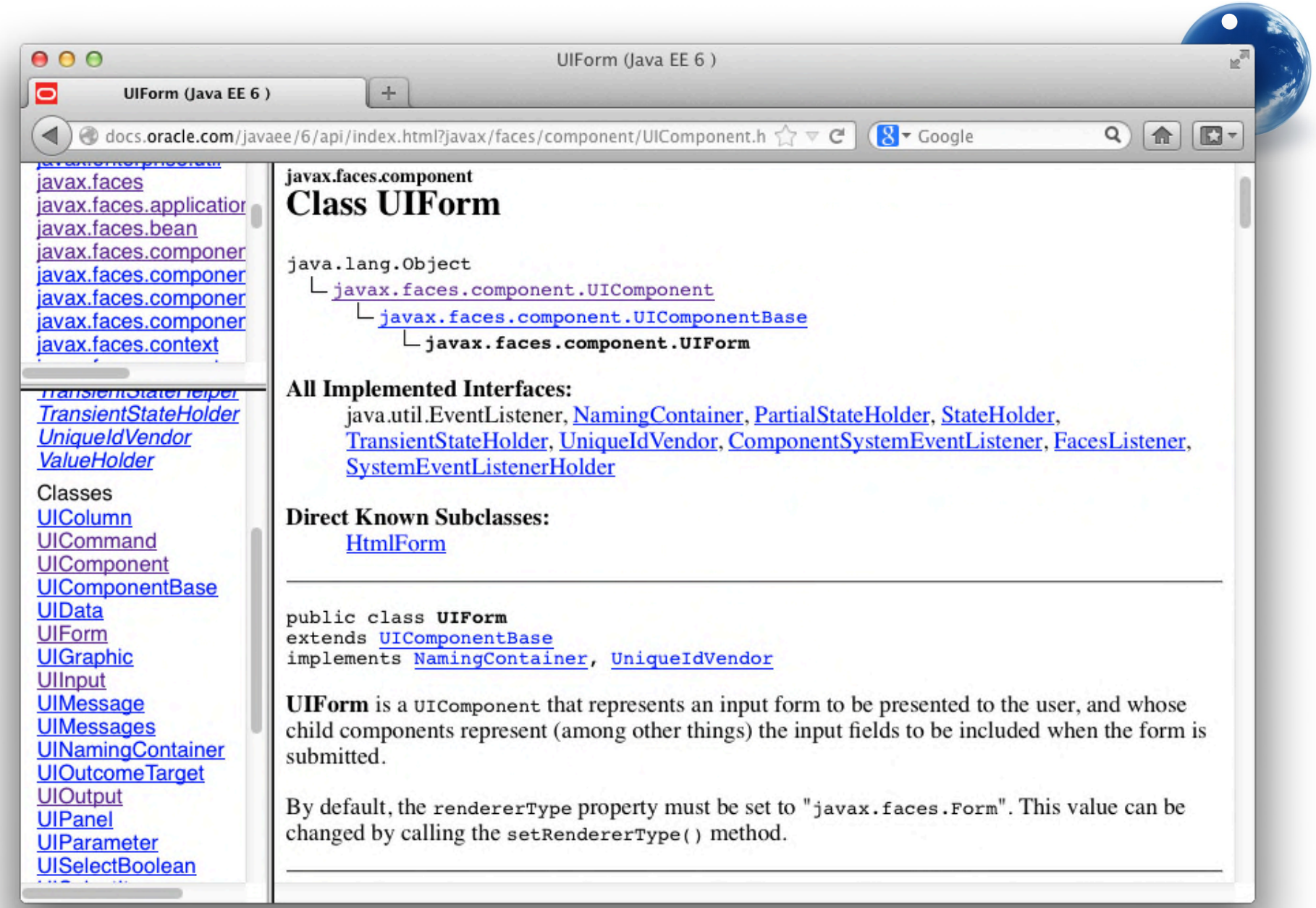


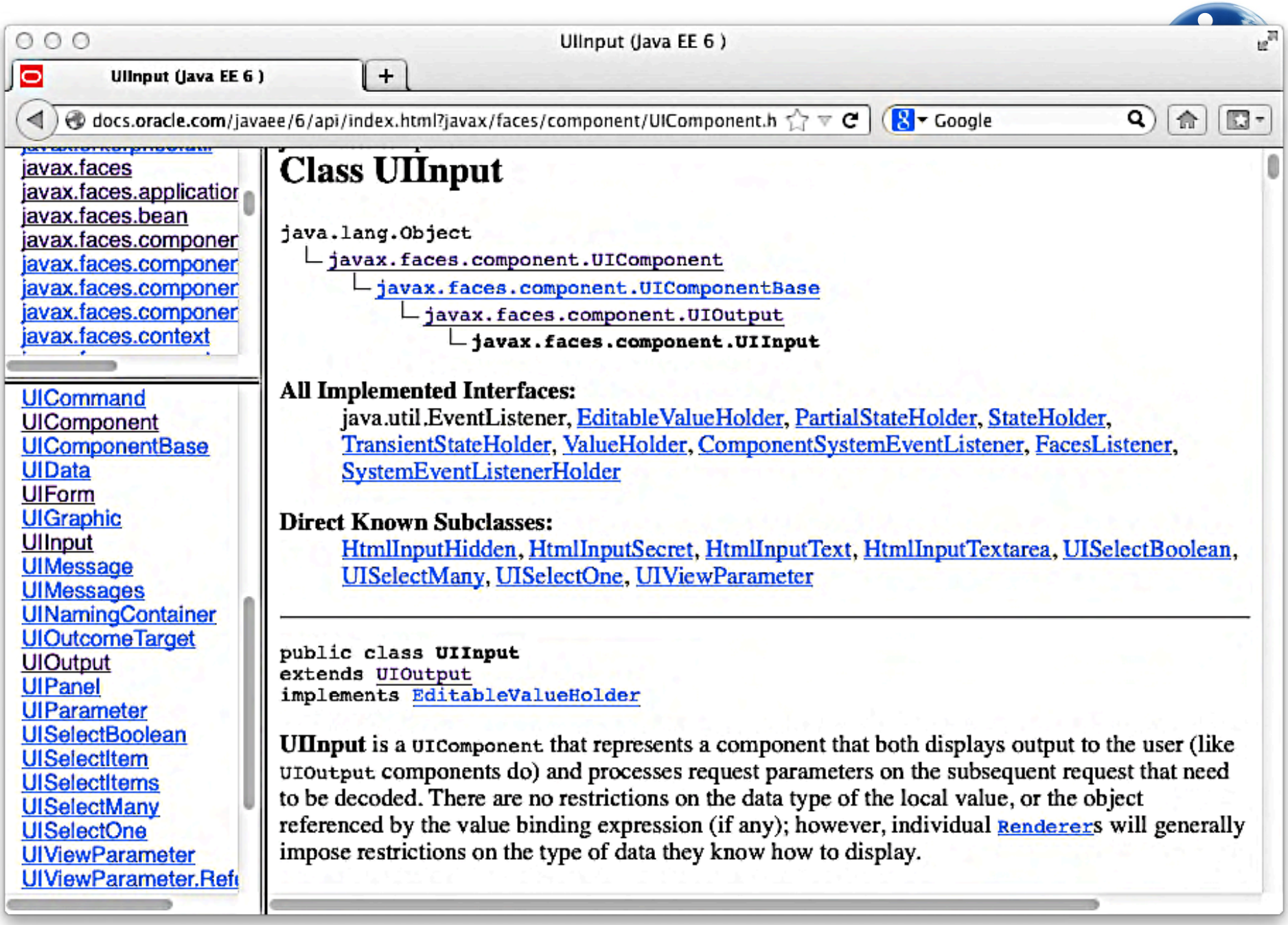
JSF UI component tree exemple

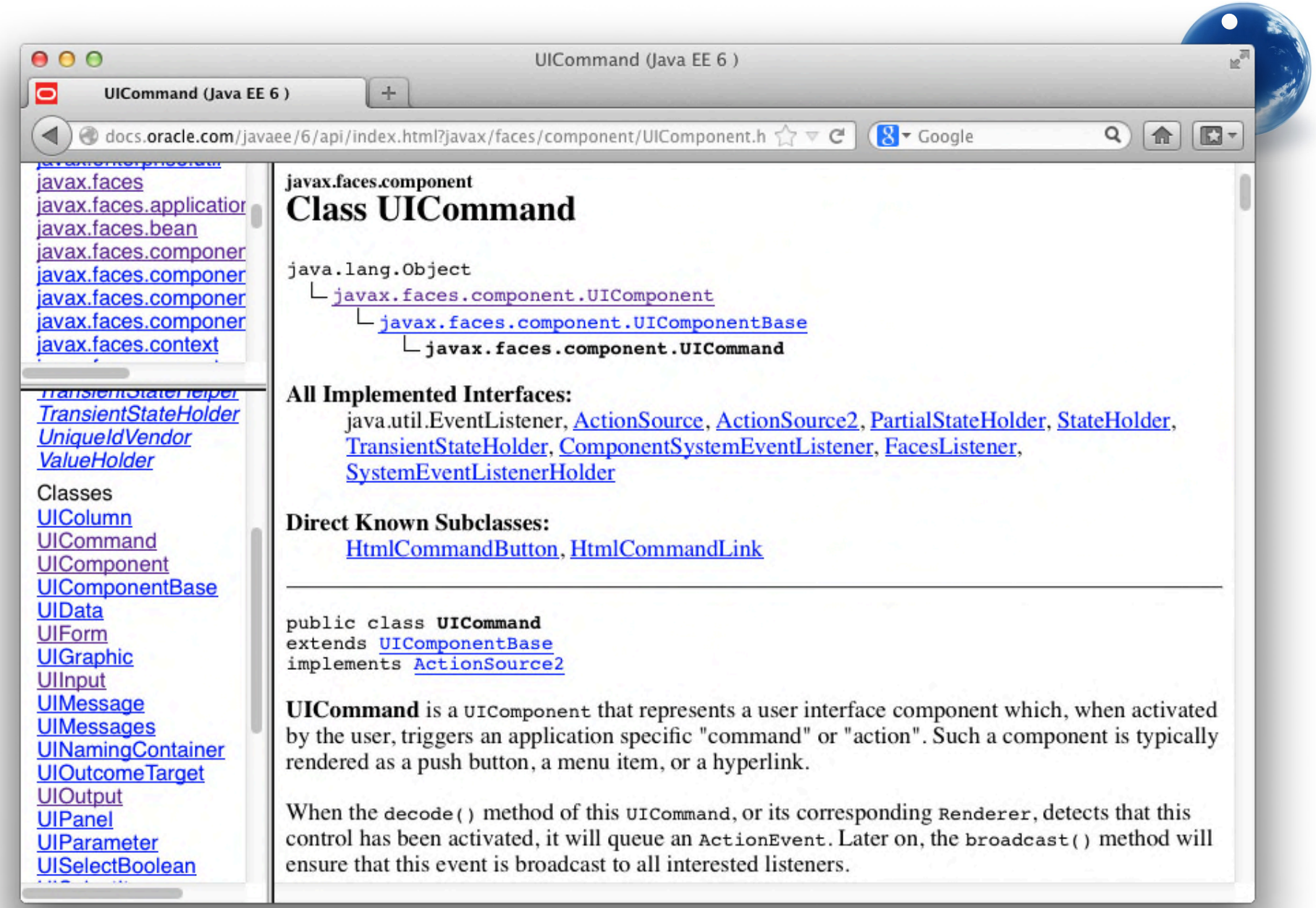



UIViewRoot (Java EE 6)
UIViewRoot (Java EE 6)
docs.oracle.com/javaee/6/api/index.html?javax/faces/component/UIComponent.h
Google
javax.faces
javax.faces.applicatior
javax.faces.bean
javax.faces.componer
javax.faces.componer
javax.faces.componer
javax.faces.componer
javax.faces.context
UIGraphic
UIInput
UIMessage
UIMessages
UINamingContainer
UIOutcomeTarget
UIOutput
UIPanel
UIParameter
UISelectBoolean
UISelectItem
UISelectItems
UISelectMany
UISelectOne
UIViewParameter
UIViewParameter.Refe
UIViewRoot
Exceptions
UpdateModelExceptio
Annotation Types
Class UIViewRoot
java.lang.Object
javax.faces.component.UIComponent
javax.faces.component.UIComponentBase
javax.faces.component.UIViewRoot
All Implemented Interfaces:
java.util.EventListener, PartialStateHolder, StateHolder, TransientStateHolder, UniqueIdVendor, ComponentSystemEventListener, FacesListener, SystemEventListenerHolder
public class UIViewRoot
extends UIComponentBase
implements UniqueIdVendor
UIViewRoot is the UIComponent that represents the root of the UIComponent tree. This component renders markup as the response to Ajax requests. It also serves as the root of the component tree, and as a place to hang per-view PhaseListeners.
For each of the following lifecycle phase methods:
processDecodes(javax.faces.context.FacesContext)
processValidators(javax.faces.context.FacesContext)
processUpdates(javax.faces.context.FacesContext)


UIOutput (Java EE 6)
UIOutput (Java EE 6)
docs.oracle.com/javaee/6/api/index.html?javax/faces/component/UIComponent.h
Google
javax.faces
javax.faces.application
javax.faces.bean
javax.faces.componer
javax.faces.componer
javax.faces.componer
javax.faces.componer
javax.faces.context
UIComponent
UIComponentBase
UIData
UIForm
UIGraphic
UIInput
UIMessage
UIMessages
UINamingContainer
UIOutcomeTarget
UIOutput
UIPanel
UIParameter
UISelectBoolean
UISelectItem
UISelectItems
UISelectMany
UISelectOne
UIViewParameter
UIViewParameter.Ref
UIViewRoot
javax.faces.component
Class UIOutput
java.lang.Object
javax.faces.component.UIComponent
javax.faces.component.UIComponentBase
javax.faces.component.UIOutput
All Implemented Interfaces:
java.util.EventListener, PartialStateHolder, StateHolder, TransientStateHolder, ValueHolder, ComponentSystemEventListener, FacesListener, SystemEventListenerHolder
Direct Known Subclasses:
HtmlBody, HtmlDoctype, HtmlHead, HtmlOutputFormat, HtmlOutputLabel, HtmlOutputLink, HtmlOutputText, UIInput, UIOutcomeTarget
public class UIOutput
extends UIComponentBase
implements ValueHolder
UIOutput is a UIComponent that has a value, optionally retrieved from a model tier bean via a value expression, that is displayed to the user. The user cannot directly modify the rendered value; it is for display purposes only.
During the Render Response phase of the request processing lifecycle, the current value of this component must be converted to a String (if it is not already), according to the following rules:


UIForm (Java EE 6)
UIForm (Java EE 6)
docs.oracle.com/javaee/6/api/index.html?javax/faces/component/UIComponent.h
Google
javax.faces
javax.faces.application
javax.faces.bean
javax.faces.componer
javax.faces.componer
javax.faces.componer
javax.faces.componer
javax.faces.context
TransientStateHolder
UniqueIdVendor
ValueHolder
Classes
UIColumn
UICommand
UIComponent
UIComponentBase
UIData
UIForm
UIGraphic
UIInput
UIMessage
UIMessages
UINamingContainer
UIOutcomeTarget
UIOutput
UIPanel
UIParameter
UISelectBoolean
javax.faces.component
Class UIForm
java.lang.Object
javax.faces.component.UIComponent
javax.faces.component.UIComponentBase
javax.faces.component.UIForm
All Implemented Interfaces:
java.util.EventListener, NamingContainer, PartialStateHolder, StateHolder, TransientStateHolder, UniqueIdVendor, ComponentSystemEventListener, FacesListener, SystemEventListenerHolder
Direct Known Subclasses:
HtmlForm
public class UIForm
extends UIComponentBase
implements NamingContainer, UniqueIdVendor
UIForm is a UIComponent that represents an input form to be presented to the user, and whose child components represent (among other things) the input fields to be included when the form is submitted.
By default, the rendererType property must be set to "javax.faces.Form". This value can be changed by calling the setRendererType() method.

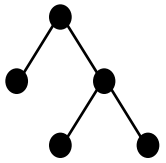

UIInput (Java EE 6)
UIInput (Java EE 6)
docs.oracle.com/javaee/6/api/index.html?javax/faces/component/UIComponent.h
Google
javax.faces
javax.faces.applicatior
javax.faces.bean
javax.faces.componer
javax.faces.componer
javax.faces.componer
javax.faces.componer
javax.faces.context
UICommand
UIComponent
UIComponentBase
UIData
UIForm
UIGraphic
UIInput
UIMessage
UIMessages
UINamingContainer
UIOutcomeTarget
UIOutput
UIPanel
UIParameter
UISelectBoolean
UISelectItem
UISelectItems
UISelectMany
UISelectOne
UIViewParameter
UIViewParameter.Refe
Class UIInput
java.lang.Object
javax.faces.component.UIComponent
javax.faces.component.UIComponentBase
javax.faces.component.UIOutput
javax.faces.component.UIInput
All Implemented Interfaces:
java.util.EventListener, EditableValueHolder, PartialStateHolder, StateHolder, TransientStateHolder, ValueHolder, ComponentSystemEventListener, FacesListener, SystemEventListenerHolder
Direct Known Subclasses:
HtmlInputHidden, HtmlInputSecret, HtmlInputText, HtmlInputTextarea, UISelectBoolean, UISelectMany, UISelectOne, UIViewParameter
public class UIInput
extends UIOutput
implements EditableValueHolder
UIInput is a UIComponent that represents a component that both displays output to the user (like UIOutput components do) and processes request parameters on the subsequent request that need to be decoded. There are no restrictions on the data type of the local value, or the object referenced by the value binding expression (if any); however, individual Renderers will generally impose restrictions on the type of data they know how to display.


UICommand (Java EE 6)
UICommand (Java EE 6)
docs.oracle.com/javaee/6/api/index.html?javax/faces/component/UIComponent.h
Google
javax.faces
javax.faces.applicatior
javax.faces.bean
javax.faces.componer
javax.faces.componer
javax.faces.componer
javax.faces.componer
javax.faces.context
TransientStateHolder
UniqueIdVendor
ValueHolder
Classes
UIColumn
UICommand
UIComponent
UIComponentBase
UIData
UIForm
UIGraphic
UIInput
UIMessage
UIMessages
UINamingContainer
UIOutcomeTarget
UIOutput
UIPanel
UIParameter
UISelectBoolean
javax.faces.component
Class UICommand
java.lang.Object
javax.faces.component.UIComponent
javax.faces.component.UIComponentBase
javax.faces.component.UICommand
All Implemented Interfaces:
java.util.EventListener, ActionSource, ActionSource2, PartialStateHolder, StateHolder, TransientStateHolder, ComponentSystemEventListener, FacesListener, SystemEventListenerHolder
Direct Known Subclasses:
HtmlCommandButton, HtmlCommandLink
public class UICommand
extends UIComponentBase
implements ActionSource2
UICommand is a UIComponent that represents a user interface component which, when activated by the user, triggers an application specific "command" or "action". Such a component is typically rendered as a push button, a menu item, or a hyperlink.
When the decode() method of this UICommand, or its corresponding Renderer, detects that this control has been activated, it will queue an ActionEvent. Later on, the broadcast() method will ensure that this event is broadcast to all interested listeners.

Sun Tutorial

- UICommand:
  - Represents a control that fires actions when activated.
- UIInput:
  - Takes data input from a user. This class is a subclass of UIOutput.
- UIOutput:
  - Displays data output on a page.
- UIViewRoot:
  - Represents the root of the component tree.

http://docs.oracle.com/javaee/6/tutorial/doc/bnaqd.html



JSF Managed bean

- Managed Beans is a java bean managed by the JSF framework.
- Managed bean is a lightweight container-managed object.
- Components in a page are associated with managed beans that provide application logic.
- Managed Bean is a regular Java Bean class registered with JSF.

JSF Managed bean properties

- A managed bean is created with a constructor with no arguments, a set of properties, and a set of methods that perform functions for a component.
- Each of the managed bean properties can be bound to one of the following:
  - A component value
  - A component instance
  - A converter instance
  - A listener instance
  - A validator instance

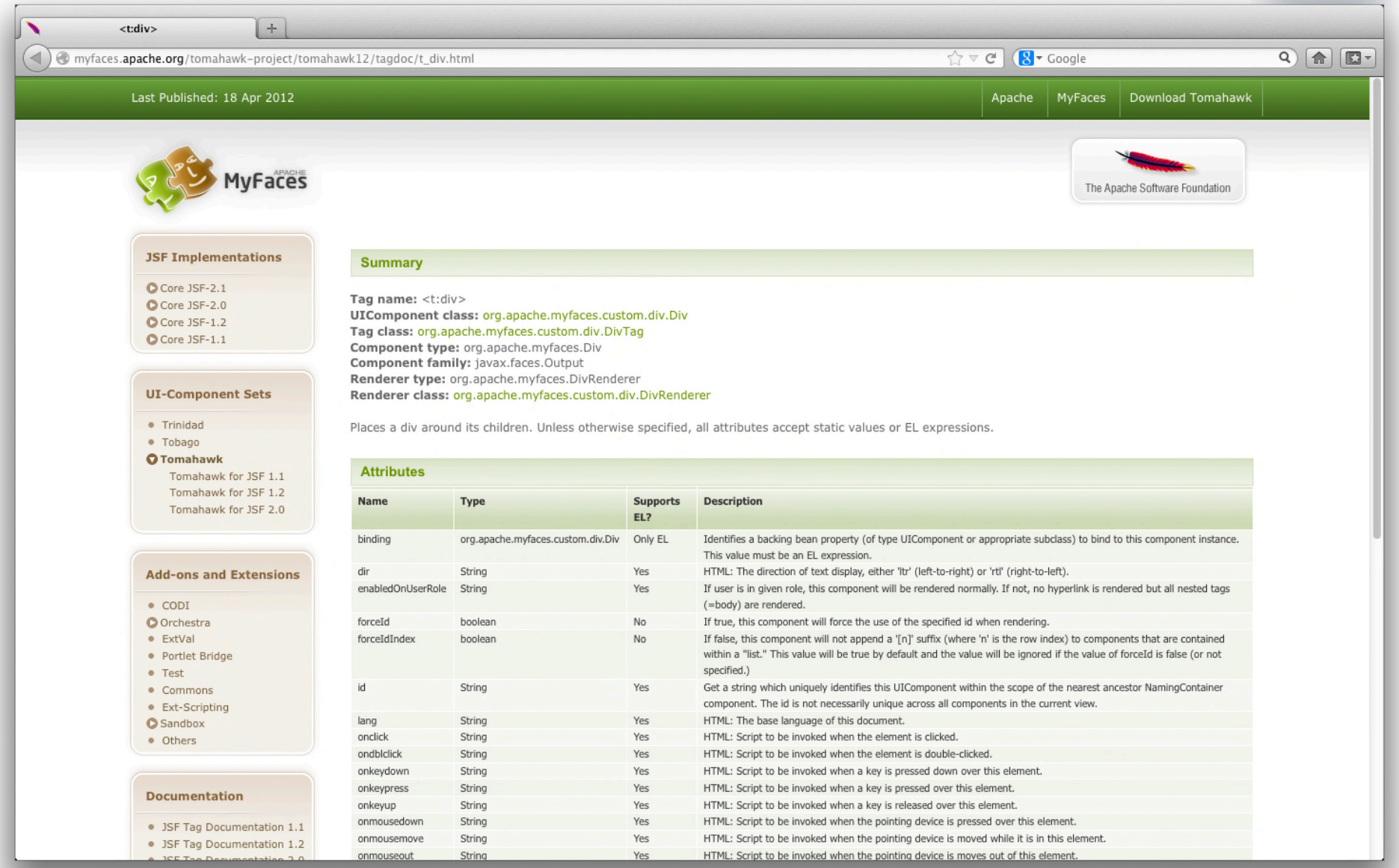
HelloJSF managed bean

```
package helloJSF;

import java.io.Serializable;

public class HelloJSFbean  implements Serializable {

	private static final long serialVersionUID = 1L;

	private String name;

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}
}
```

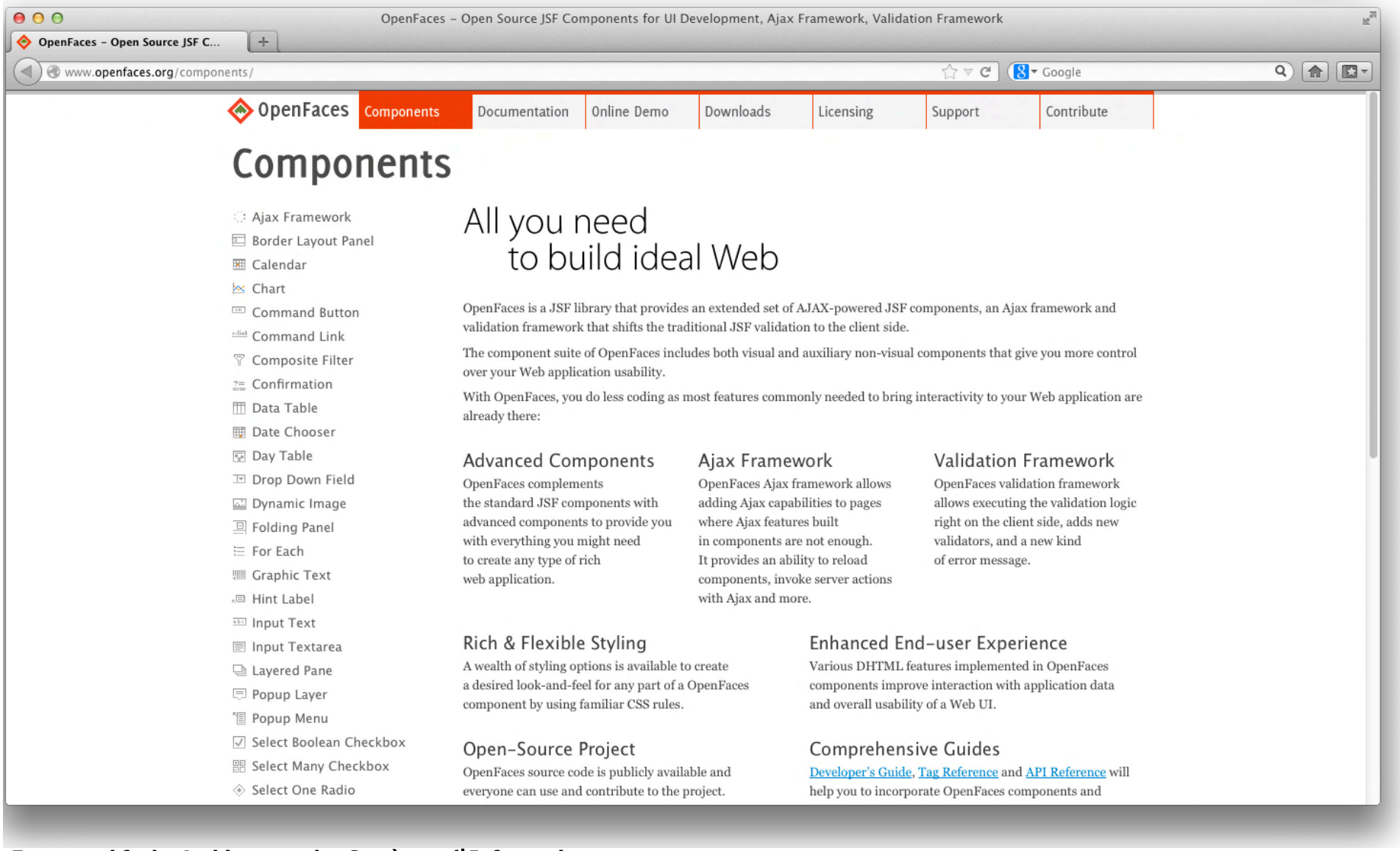
3rd Party JSF Components

- MyFaces
- ICEfaces
- RichFaces
- jquery4jsf
- OpenFaces
- PrimeFaces

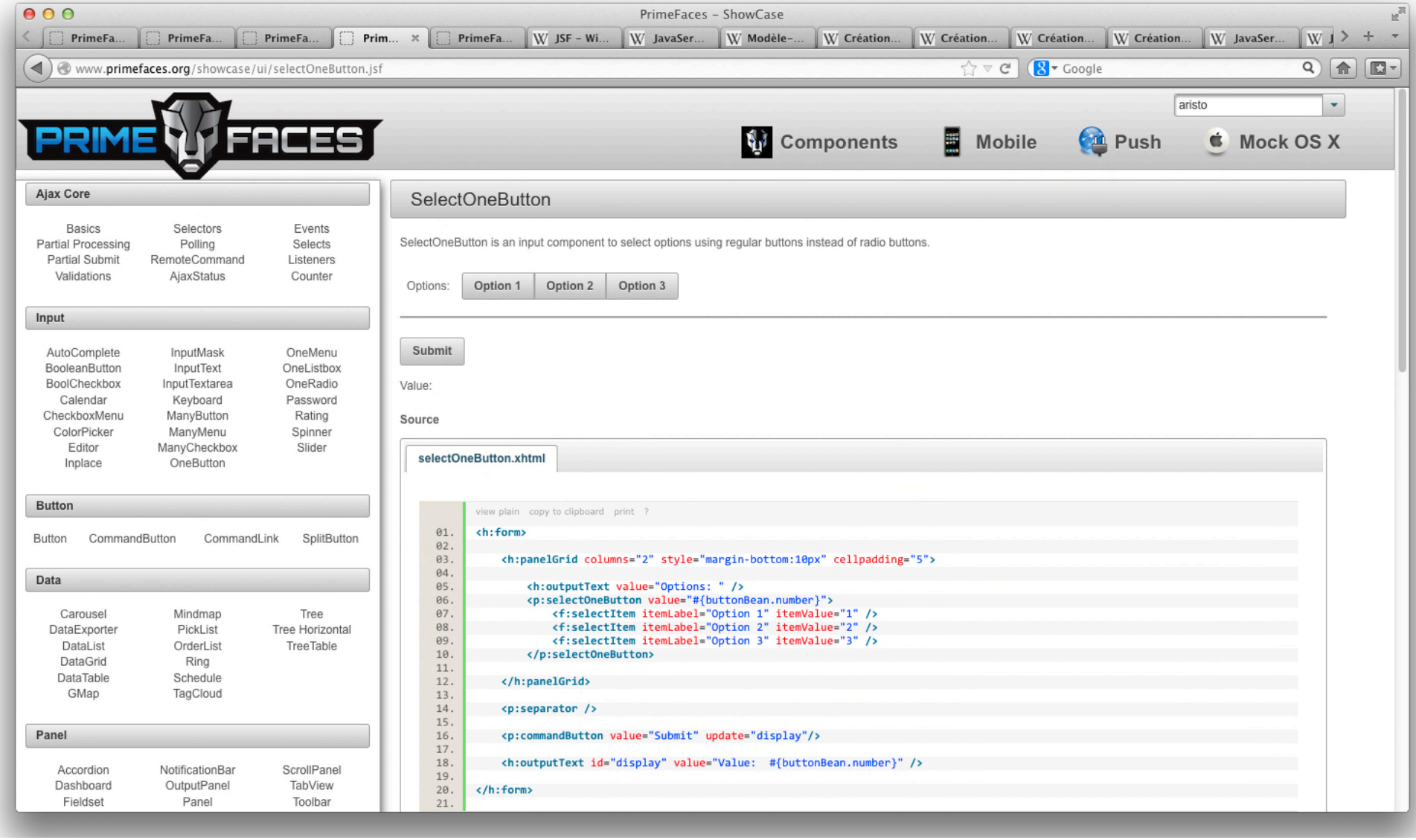
MyFaces


<t:div>
myfaces.apache.org/tomahawk-project/tomahawk12/tagdoc/t_div.html
Google
Last Published: 18 Apr 2012
Apache
MyFaces
Download Tomahawk
MyFaces
The Apache Software Foundation
JSF Implementations
Core JSF-2.1
Core JSF-2.0
Core JSF-1.2
Core JSF-1.1
UI-Component Sets
Trinidad
Tobago
Tomahawk
Tomahawk for JSF 1.1
Tomahawk for JSF 1.2
Tomahawk for JSF 2.0
Add-ons and Extensions
CODI
Orchestra
ExtVal
Portlet Bridge
Test
Commons
Ext-Scripting
Sandbox
Others
Documentation
JSF Tag Documentation 1.1
JSF Tag Documentation 1.2
Summary
Tag name: <t:div>
UIComponent class: org.apache.myfaces.custom.div.Div
Tag class: org.apache.myfaces.custom.div.DivTag
Component type: org.apache.myfaces.Div
Component family: javax.faces.Output
Renderer type: org.apache.myfaces.DivRenderer
Renderer class: org.apache.myfaces.custom.div.DivRenderer
Places a div around its children. Unless otherwise specified, all attributes accept static values or EL expressions.
Attributes
Name | Type | Supports EL? | Description
binding | org.apache.myfaces.custom.div.Div | Only EL | Identifies a backing bean property (of type UIComponent or appropriate subclass) to bind to this component instance. This value must be an EL expression.
dir | String | Yes | HTML: The direction of text display, either 'ltr' (left-to-right) or 'rtl' (right-to-left).
enabledOnUserRole | String | Yes | If user is in given role, this component will be rendered normally. If not, no hyperlink is rendered but all nested tags (=body) are rendered.
forceId | boolean | No | If true, this component will force the use of the specified id when rendering.
forceIdIndex | boolean | No | If false, this component will not append a '[n]' suffix (where 'n' is the row index) to components that are contained within a "list." This value will be true by default and the value will be ignored if the value of forceId is false (or not specified.)
id | String | Yes | Get a string which uniquely identifies this UIComponent within the scope of the nearest ancestor NamingContainer component. The id is not necessarily unique across all components in the current view.
lang | String | Yes | HTML: The base language of this document.
onclick | String | Yes | HTML: Script to be invoked when the element is clicked.
ondblclick | String | Yes | HTML: Script to be invoked when the element is double-clicked.
onkeydown | String | Yes | HTML: Script to be invoked when a key is pressed down over this element.
onkeypress | String | Yes | HTML: Script to be invoked when a key is pressed over this element.
onkeyup | String | Yes | HTML: Script to be invoked when a key is released over this element.
onmousedown | String | Yes | HTML: Script to be invoked when the pointing device is pressed over this element.
onmousemove | String | Yes | HTML: Script to be invoked when the pointing device is moved while it is in this element.
onmouseout | String | Yes | HTML: Script to be invoked when the pointing device is moves out of this element.

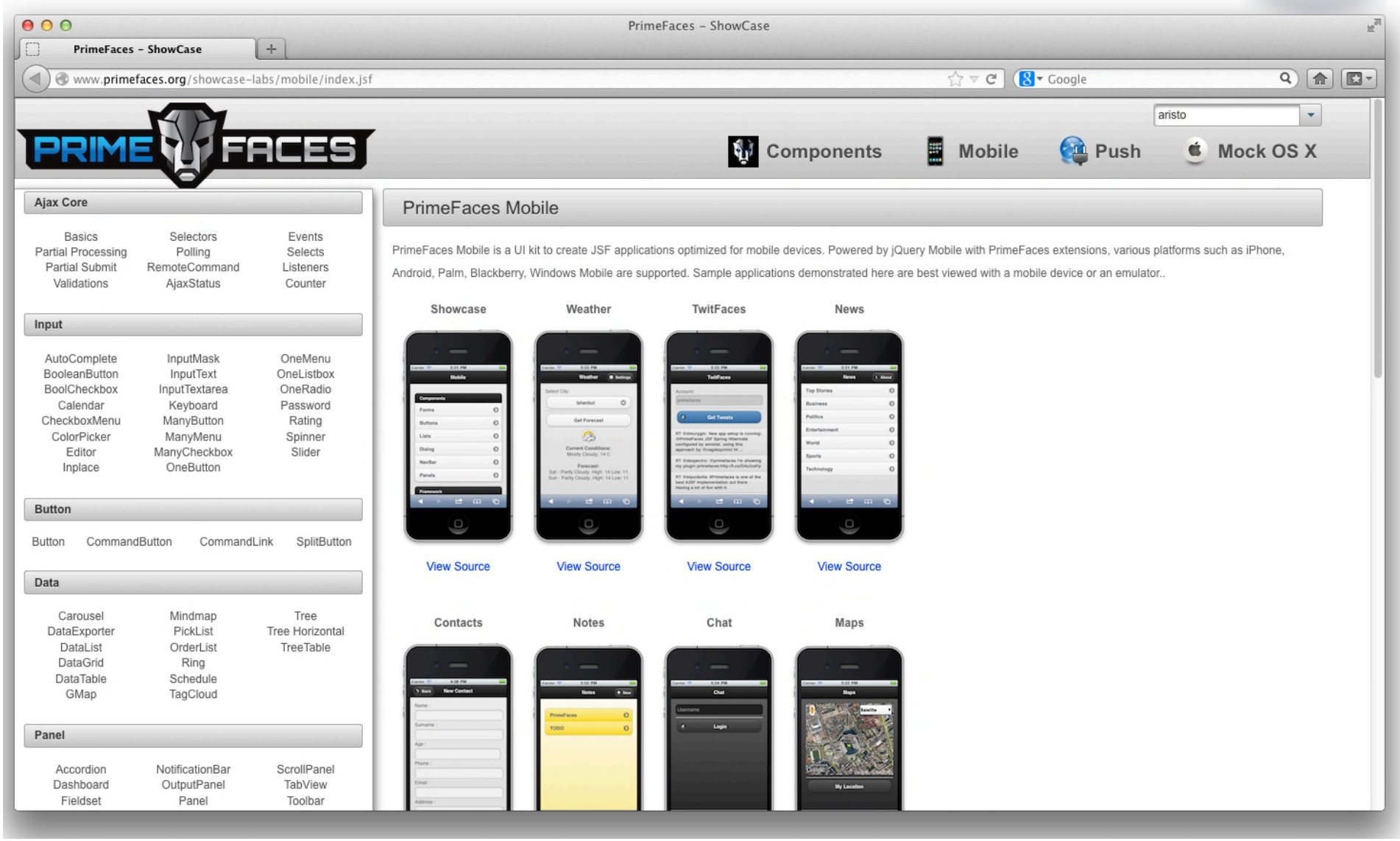
OpenFaces


OpenFaces – Open Source JSF Components for UI Development, Ajax Framework, Validation Framework
OpenFaces – Open Source JSF C...
www.openfaces.org/components/
Google
OpenFaces
Components
Documentation
Online Demo
Downloads
Licensing
Support
Contribute
Components
Ajax Framework
Border Layout Panel
Calendar
Chart
Command Button
Command Link
Composite Filter
Confirmation
Data Table
Date Chooser
Day Table
Drop Down Field
Dynamic Image
Folding Panel
For Each
Graphic Text
Hint Label
Input Text
Input Textarea
Layered Pane
Popup Layer
Popup Menu
Select Boolean Checkbox
Select Many Checkbox
Select One Radio
All you need to build ideal Web
OpenFaces is a JSF library that provides an extended set of AJAX-powered JSF components, an Ajax framework and validation framework that shifts the traditional JSF validation to the client side.
The component suite of OpenFaces includes both visual and auxiliary non-visual components that give you more control over your Web application usability.
With OpenFaces, you do less coding as most features commonly needed to bring interactivity to your Web application are already there:
Advanced Components
OpenFaces complements the standard JSF components with advanced components to provide you with everything you might need to create any type of rich web application.
Ajax Framework
OpenFaces Ajax framework allows adding Ajax capabilities to pages where Ajax features built in components are not enough. It provides an ability to reload components, invoke server actions with Ajax and more.
Validation Framework
OpenFaces validation framework allows executing the validation logic right on the client side, adds new validators, and a new kind of error message.
Rich & Flexible Styling
A wealth of styling options is available to create a desired look-and-feel for any part of a OpenFaces component by using familiar CSS rules.
Enhanced End-user Experience
Various DHTML features implemented in OpenFaces components improve interaction with application data and overall usability of a Web UI.
Open-Source Project
OpenFaces source code is publicly available and everyone can use and contribute to the project.
Comprehensive Guides
Developer's Guide, Tag Reference and API Reference will help you to incorporate OpenFaces components and

PrimeFaces Show cases



PrimeFaces Show cases



PrimeFaces Show cases