PrimeFaces: Overview, Installation, and Setup

Originals of slides and source code for examples: http://www.coreservlets.com/JSF-Tutorial/primefaces/
Also see the JSF 2 tutorial – http://www.coreservlets.com/JSF-Tutorial/jsf2/
and customized JSF2 and PrimeFaces training courses – http://courses.coreservlets.com/jsf-training.html

Customized Java EE Training: http://courses.coreservlets.com/
Java 7, Java 8, JSF 2, PrimeFaces, Android, JSP, Ajax, jQuery, Spring MVC, RESTful Web Services, GWT, Hadoop
Developed and taught by well-known author and developer. At public venues or onsite at your location.

For live training on JSF 2, PrimeFaces, or other Java EE topics, email hall@coreservlets.com
Marty is also available for consulting and development support

Taught by the author of *Core Servlets and JSP*, this tutorial, and JSF 2.2 version of *Core JSF*. Available at public venues, or customized versions can be held on-site at your organization.

- Courses developed and taught by Marty Hall
  - JSF 2, PrimeFaces, Ajax, jQuery, Spring MVC, JSP, Android, general Java, Java 8 lambdas/streams, GWT, custom topic mix
    - Courses available in any location worldwide. Maryland/DC area companies can also choose afternoon/evening courses.
- Courses developed and taught by coreservlets.com experts (edited by Marty)
  - Hadoop, Spring, Hibernate/JPA, RESTful Web Services

Contact hall@coreservlets.com for details
Topics in This Section

- Third-Party JSF component libraries
- Overview of PrimeFaces
- Installation and setup
- Simple examples
  - With and without server behavior
Component Libraries: Idea

- **Component libraries do**
  - Add rich GUI elements, for both input and output
    - Sliders, autocomplete textfields, dialog boxes, tabbed panels, color pickers, masked input fields, etc.
- **Component libraries do not**
  - Require you to use a particular Java server (e.g., JBoss)
  - Change the general way you write your Java code
  - Change the JSF flow of control
  - Change the overall structure of facelets pages
  - Prevent you from using *any* of the techniques covered in the earlier JSF 2 tutorial
  - Bottom line
    - They add GUI elements, but are not different frameworks

Popular JSF 2 Component Libraries

- **PrimeFaces**
  - [http://primefaces.org/](http://primefaces.org/)
- **RichFaces**
  - [http://www.jboss.org/richfaces](http://www.jboss.org/richfaces)
- **IceFaces**
- **Oracle ADF Faces**
- **OpenFaces**
  - [http://openfaces.org/](http://openfaces.org/)
- **Tomahawk**
  - [http://myfaces.apache.org/tomahawk/index.html](http://myfaces.apache.org/tomahawk/index.html)
- **Trinidad**
  - [http://myfaces.apache.org/trinidad/index.html](http://myfaces.apache.org/trinidad/index.html)
Usage Trends: Job Postings

Usage Trends: Google Searches
Overview of PrimeFaces

Customized Java EE Training: http://courses.coreservlets.com/
Java 7, Java 8, JSF 2, PrimeFaces, Android, JSP, Ajax, jQuery, Spring MVC, RESTful Web Services, GWT, Hadoop
Developed and taught by well-known author and developer. At public venues or onsite at your location.

General Features

• **Rich visual components**
  – Calendars, charts, sliders, image viewers, much more
    • Many are based on jQuery UI and jQuery plugins

• **Completely compliant with JSF 2**
  – Runs in any JSF 2 implementation

• **Free and open source**
  – Free for all apps, including commercial ones. Apache license.

• **Lightweight**
  – Very low overhead to adding PrimeFaces components to existing JSF 2 apps

• **Optional theming**
  – Rich skinning options, with interactive theme builder.
    • But can easily use components in existing JSF 2 app.
Types of Components

- **Input elements**
  - Popup calendar, slider, autocompleter, masked fields, etc.
- **Popup windows**
  - Dialog boxes, menus, Mac-like “Growl” notifications, etc
- **Grouping panels**
  - Accordion panel, tab panel, scroll panel, dashboard, etc.
- **Charts**
  - Bar charts, pie charts, line graphs, etc.
- **Image viewers**
  - Many types of image galleries and viewers
- **Skins (themes) and interactive skin builder**
- **Rich Ajax support**
  - Built into many components
- **Drag and drop API**

Installation and Setup
Basic Installation

- **Most components**
  - Download JAR file
    - [http://www.primefaces.org/downloads.html](http://www.primefaces.org/downloads.html)
    - Already included in app that goes with this tutorial
  - Put JAR file in WEB-INF/lib folder of your application
    - Same place all other JAR files go
  - Add PrimeFaces declaration to `<html ...>` start tag
    - `<html ... xmlns:p="http://primefaces.org/ui">`
  - Start using PrimeFaces tags. (It is really that easy!)
    - E.g., put `<p:calendar/>` inside any JSF 2 form

- **A few specialized components**
  - PDF exporter, Excel exporter, file upload, and Ajax push require some extra JAR files.
    - Well documented in User Manual

Installation and Setup

- **Put JAR file in app**
- **Add PrimeFaces declaration**
  - `<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:h="http://xmlns.jcp.org/jsf/html"
    xmlns:p="http://primefaces.org/ui">`

- **Start using PrimeFaces components**
  - That’s really all there is to it!
Documentation

- **User’s Guide**
  - [http://www.primefaces.org/documentation.html](http://www.primefaces.org/documentation.html)
    - Moderately good by open source standards
  - Download PDF of Guide to your local computer
    - This is only documentation most developers use
    - But main page above also has link to JavaDoc for underlying Java code, which is used once in a while

- **Eclipse code completion and palette**
  - Automatic in projects that have PrimeFaces JAR file, as long as JSF project facet enabled
    - JSF facet described in earlier tutorial on JSF 2 setup

- **Showcase**
  - [http://www.primefaces.org/showcase/](http://www.primefaces.org/showcase/)
    - Has easy-to-copy facelets and Java code samples for each example

---

Simple Example: No Server Behavior
Simple Example  
(No Server Behavior)

- Put JAR file in app
- Added PrimeFaces declaration
  ```html
  <html xmlns="http://www.w3.org/1999/xhtml"
       xmlns:h="http://xmlns.jcp.org/jsf/html"
       xmlns:p="http://primefaces.org/ui">
  ```
- Inserted PrimeFaces components
  ```html
  <h:form>
  <b>Choose number:</b> <p:spinner/>
  <br/>
  <b>Choose date:</b> <p:calendar/>
  </h:form>
  ```

Downloaded from http://www.primefaces.org/downloads.html

Results

![Results](image-url)
Simple Example: With Server Behavior

Better but Still Simple Example (with Server Behavior)

• Point
  – PrimeFaces input components convert types automatically
    • p:spinner converts to double (or int if step is whole num)
    • p:calendar converts to Date
  – No user conversion needed
    • And no need for validatorMessage since neither one will permit the user to enter an illegal type
Bean

@ManagedBean
public class TestBean {
    private int number;
    private Date date;

    public int getNumber() {
        return number;
    }
    public void setNumber(int number) {
        this.number = number;
    }
    public Date getDate() {
        return date;
    }
    public void setDate(Date date) {
        this.date = date;
    }
    public String doAction() {
        return "show-test-data";
    }
}

Input Form (test.xhtml)

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
     xmlns:h="http://xmlns.jcp.org/jsf/html"
     xmlns:p="http://primefaces.org/ui">
    <h:head>...</h:head>
    <h:body>
        ...
        <h:form>
            <b>Choose number:</b> <p:spinner value="#{testBean.number}"/>
            <br/>
            <b>Choose date:</b> <p:calendar value="#{testBean.date}"/>
            <br/>
            <p:commandButton action="#{testBean.doAction}" ajax="false"
                value="Submit"/>
        </h:form>
        ...
    </h:body></html>

It works fine to use h:commandButton, but the advantage of p:commandButton is that the button picks up the fonts, sizes, and colors of the current theme. But, if you use p:commandButton, you have to use ajax="false" if you want it to behave just like h:commandButton. In general, it works fine to mix p:blah and h:blah elements in the same form, but only the p:blah elements will automatically follow the PrimeFaces theme.
Results Page
(show-test-data.xhtml)

```html
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
xmlns:h="http://xmlns.jcp.org/jsf/html">
<h:head><title>Test Data</title></h:head>
<h:body>
<h1 class="ui-widget-header ui-corner-all" align="center">Test Data</h1>
<p/>
<ul>
<li>Selected number: #{testBean.number}</li>
<li>Selected date: #{testBean.date}</li>
</ul>
</h:body></html>
```

Results
Wrap-Up

Summary

• **Component libraries**
  – Add rich GUI elements for input or output
  – Normal JSF code and approaches still work
    • Extra GUI elements, not new framework!

• **Popular component libraries**
  – All of PrimeFaces, RichFaces, IceFaces, Tomahawk, and Oracle ADF Faces are widely used
  – But PrimeFaces is (arguably) the least intrusive and most widely used

• **Installation**
  – Drop in JAR file, add declaration to <html…>, go!
Questions?

More info:
- http://www.coreservlets.com/jsf-tutorial/2/ - JSF 2.2 tutorial
- http://www.coreservlets.com/ - JSF 2, PrimeFaces, Java 7 or 8, Ajax, jQuery, Hadoop, RESTful Web Services, Android, HTML5, Spring, Hibernate, Services, JSP, GWT, and other Java EE training

Customized Java EE Training: http://courses.coreservlets.com/
Java 7, Java 8, JSF 2, PrimeFaces, Android, JSP, Ajax, jQuery, Hadoop, RESTful Web Services, GWT, Hadoop
Developed and taught by well-known author and developer. At public venues or onsite at your location.