# ELAB JSF

## Overview

JSF ist eine Komponente der Java EE Standards, vorgesehen für die schnelle Entwicklung von Web Anwendungen. Es trennt die Darstellungsschicht von der Programmlogik, jedoch etwas anders als beim Modell 2 (MVC Pattern).

Die JSF Template Engine erzeugt HTML Seiten auf Basis von Vorlagen-Dateien, indem sie Platzhalter mit Daten aus Java Beans ausfüllt und Methoden aufgrund von Ereignissen (z.B. Klick auf einen Button) aufruft. Beispiel für so einen Platzhalter:

#{user.name}

Facelets sind Templates, also Vorlagen für Webseiten, typischerweise mit der Dateiendung \*.xhtml. Facelets enthalten neben HTML auch Ausdrücke aus der Expression Language und JSF Tags. Expression Language Ausdrücke sehen so aus:

#{user.name}

#{geld.eingang - geld.ausgang}

#{auto.farbe != rot}

Zum Aufbau von bedingten Abschnitten, Schleifen und anderen Kontrollstrukturen verwendet man JSF Tags. Beispiele:

<ui:fragment rendered="#{user.alter > 12}">

Du bist alt genug

</ui:fragment>

<ui:repeat var="auto" value="#{meinBean.alleAutos}">

Das Auto #{auto.name} ist #{auto.farbe}.

</ui:repeat>

Weiterhin gibt es in der JSTL Library Tags zum Erzeugen von HTML Elementen. Zum Beispiel erzeugt das h:selectBooleanCheckboxTag eine Check-Box, die an eine booleansche Variable gebunden ist. Andere Dateiformate sind theoretisch möglich, aber meines Wissens nach nicht umgesetzt worden.

JSF bietet ungefähr gleich gute Performance, wie andere Web Frameworks, also mittelmäßige. Einerseits wird die Softwareentwicklung vereinfacht, was andererseits signifikant Rechenleistung kostet. Doch Rechenleistung ist meist billiger, als Entwicklungs-Aufwand.

## JSF

### Pros

* JSF allows a UI to be composites of high level visual building blocks.
* JSF help you to build applications faster.
* It makes easy to combine complex GUIs into a single manageable component
* JSF supporting dataTable

### Cons

* Steep learning curve is one of the main dis-advantage of the JSF. However, if you are familiar with HTML and CSS concepts, it is going to be a cake walk for you
* There is no benchmarking report or promise from Sun Microsystems about the performance of JSF framework. By seeing their concept I believe it is not suitable for high performance application.
* The specification doesn’t consider bookmarking facility.
* Hardly a very few examples available for developing dynamic pages including new component and removing a component from a page based on business rule.
* Every button or link clicked results in a form post. That’s just wrong why can’t I have true links like the web is supposed to? Form submission for page navigation make complex coding for simple requirement like Cancel button. Read here to know the work around.

## JSP w /MV2

### Pros

* + Extensive Expression Language
  + Supports taglibs
  + Quicker to build simple applications

### Cons

* + No pre-implemented components
  + Translation Phase: o First request causes noticeable delay o Difficulties debugging as Servlet code is not directly written by programmer

## Access a bean

<h: inputText value=”#{book.title}”></h:inputText>