Web Frameworks
Java Server Faces

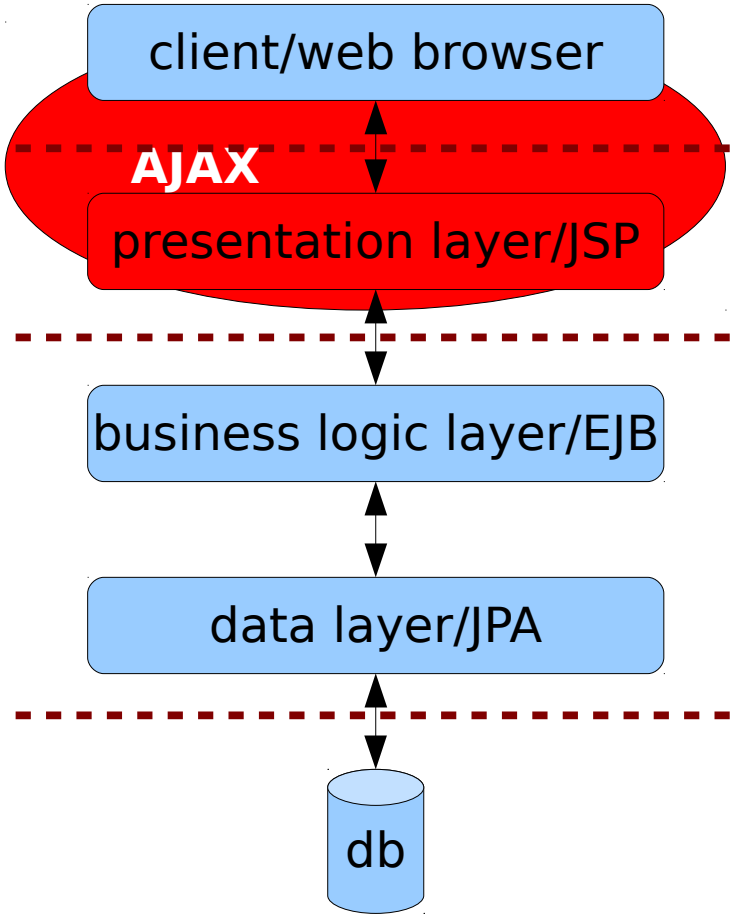
Petr Aubrecht
CA

Co dnes probereme

- AJAX rulez!!!
- Web frameworks – proč, za kolik a které
- JSF
  - důvody
  - backing bean
  - navigace
  - atd. atd.

Vždyť už všechno umíte

- Víte, jak se připojit do databáze.
- Víte, kam dát business logic.
- Víte, jak data prezentovat.
- Co ještě byste chtěli vědět?



Jak udělat ten web jednoduše!
Třeba jako SWING – nakreslit, naklikat a spustit.

Proč webové frameworky

- Servlety nás odstínily od HTTP.
- JSP nás odstínilo od servletů (starost o stream).
- Jak se oprostit od starosti o formuláře a jejich ukládání, repopulaci při opravě, validaci, hlášení chyb?
- Jak zvládnout ty cool aplikace, co po nás šéfové chtějí?
- Kolik lidí tady zná dobře Javascript, aby napsal něco a la Google Apps?
- AJAX sice rulez, ale je to STRAŠLIVĚ obtížné!

Struts

- Nejslavnější (protože první široce přijatý) frameworkem se staly Struts. Nabídly backing beanu, kterou samy naplnily hodnotami z formuláře (podle shodných jmen fieldů/atributů), nabídly validaci (dost nízkoúrovňovou) a navigaci.
- Bylo potřeba ke každému formuláři psát backing beanu, což vedlo k poměrně mnoha třídám.

Google Web Toolkit

- Existují dva velmi slavné frameworky, o kterých si nebudeme vyprávět, ale zmínka by měla padnout.
- Google Web Toolkit je založen na nápadu napsat logiku v Javě. Obsahuje kompilátor, který klientskou část přeloží do JavaScriptu. Není založen na HTML, ale UI se vytváří pomocí programu (podobně jako Swing).
- GWT má vyřešenou komunikaci browser <-> server, podporuje asynchronní zprávy.
- Právě specializovaný překladač (převzatý z Eclipse) ztěžuje automatický překlad (např. night build).

Spring Framework

- Rod Johnson při psaní knihy o J2EE (2.1) uváděl mnoho příkladů, jak usnadnit programování v J2EE. Nakonec měl tolik materiálu, že ho zpracoval jako framework. Ten byl velice úspěšný a získal velkou popularitu. Hlavním cílem byla náhrada EJB.
- Spring Framework se poměrně rychle rozrostl z náhrady EJB na celý stack technologií a konkuruje dnes i JSF.
- Výhodou je, že jde o knihovny, takže pro deployment vystačíme s Tomcatem, není třeba JEE kontejner.
- Má lepší vlastnosti, ale není standardem. Tedy existuje jediná implementace (co když se Rod rozhodne dát se na malování?).

Stripes, Velocity, Struts2 a další

- Webových frameworků se vyrojil nespočet, ale výrobci nástrojů neměli možnost soustředit se na jeden. Zkusím vyjmenovat několik open source:
  - Tapestry, Struts 2, Wicket, Cocoon, Turbine, Makumba, Maverick, Echo, Rife, DWR, Stripes, SiteMesh a další
- V poslední době přišly i z příbuzných oblastí
  - Ruby on Rails, Grails, OpenLaszlo, Flex a další
- Sun přišel s JSR 127 s návrhem na standardizaci. Velcí hráči hrají nejraději s velkými hráči.

Čistě javascriptové knihovny

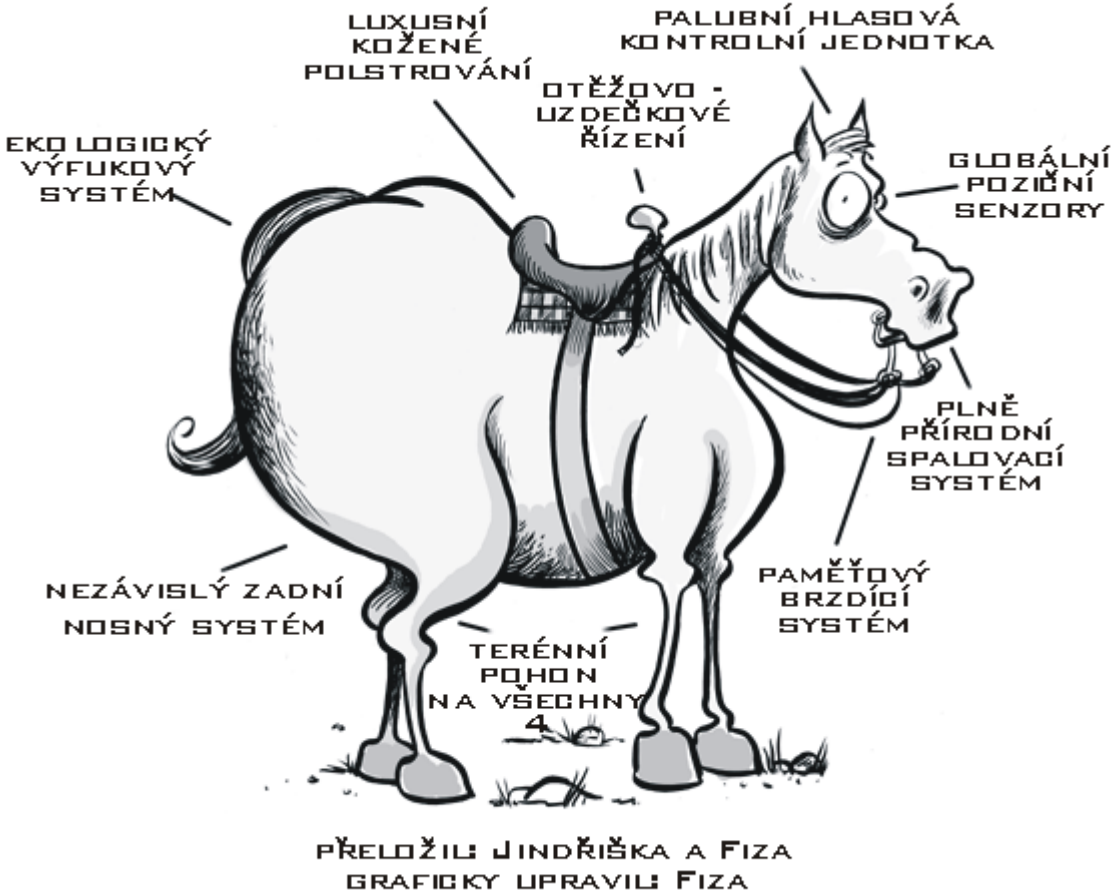
- Existují i knihovny čistě javascriptové (např. Yahoo toolkit).
- Dají se použít i pro JEE aplikace, ale je to netypické.
- Je potřeba zařídit zpracování událostí, a generování stránek ve formě vyžadované knihovnou.
- Nemají vazbu do Javy, jsou obecné a nezávislé na jazyku na serveru.
- Zde uvádím pouze pro úplnost.

Myšlenky za JSF

- pokrytí oblastí zpracované jinými frameworky
  - automatické ukládání dat z formulářů
  - repopulace formulářů
  - validace
  - zpracování chyb
  - navigace
  - šablony
- tagování (koho z vás napadlo, že Javový kód v JSP je špatně? – v JSF 2.0 lze vypnout)
- konkurovat nejen frameworkům, ale i .Netu!
- **podpora nástrojů (IDE)!**

Praktický příklad

- Co všechno je potřeba nastavit.
- Jak vypadá kód.



Co potřebujeme (1)

- záznam ve web.xml

```
<servlet>
  <servlet-name>Faces Servlet</servlet-name>
  <servlet-class>javax.faces.webapp.FacesServlet</servlet-
 class>
  <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
  <servlet-name>Faces Servlet</servlet-name>
  <url-pattern>/faces/*</url-pattern>
</servlet-mapping>
```

Co potřebujeme (2)

- konfigurace backing beans ve faces-config.xml

```
<faces-config version="1.2"
    xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
 http://java.sun.com/xml/ns/javaee/web-
 facesconfig_1_2.xsd">

<managed-bean>
  <managed-bean-name>carBack</managed-bean-name>
  <managed-bean-class>cz.eja.back.CarBack</managed-bean-
 class>
  <managed-bean-scope>session</managed-bean-scope>
</managed-bean>
```

Co potřebujeme (3a)

- konfigurace navigace ve faces-config.xml (viz také Page Flow záložka v NB)

```
<navigation-rule>
  <from-view-id>/prime/chooseprime.jsp</from-view-id>
  <navigation-case>
    <from-outcome>success</from-outcome>
    <to-view-id>/prime/nsdresult.jsp</to-view-id>
    <redirect />
  </navigation-case>
</navigation-rule>
```

Co potřebujeme (3b)

- navigace v kódu, u metody, která obsluhuje submit, vrací string, který určuje výstup
  - string se vyhledá v konfiguraci
  - null znamená žádný přesun, aplikace zůstává na té samé stránce

```
public String calc() {
   ...
   if(!ok)
        return null;
   return "success";
}
```

Co potřebujeme (4)

- struktura stránky

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib prefix="f" uri="http://java.sun.com/jsf/core"%>
<%@taglib prefix="h" uri="http://java.sun.com/jsf/html"%>
<f:view>
<html> <head> <meta ... <title ... </head>
...
  <h:form id="nsdform">
    <h:messages /> ...
  </h:form>
</body>
</html>
</f:view>
```

Jak na formuláře – EL (Expression Language)

- R/W property binding
- method call

```
<h:form id="nsdform">
  <h:inputText id="num1id" value="#{primeBack.num1}"
 validator="#{primeBack.validateNum}"/>
  <h:commandButton action="#{primeBack.calc}"
 value="calc NSD" />
</h:form>
```

Formulář a Backing Bean

```
<h:inputText id="num1id" value="#{primeBack.num1}"/>
public void setNum1(String x)
public String getNum1()

<h:commandButton action="#{primeBack.calc}"
 value="calc NSD" />
public String calc()

<h:commandButton type="submit"
 actionListener="#{primeBack.calc}" value="calc NSD" />
public void calc(ActionEvent event)
```

- nesouvisí s navigací, pouze se zavolá metoda calc

Validace

```
<h:inputText id="num1id" value="#{primeBack.num1}"
 validator="#{primeBack.validateNum}"/>


public void validateNum(FacesContext context, UIComponent
 component, Object value)
      throws ValidatorException {
...
FacesContext.getCurrentInstance().addMessage(null, new
 FacesMessage("Unable to convert to number!"));
...
throw new ValidatorException(new FacesMessage("Number must
 be >0"));
```

Zobrazení zpráv/chyb

```
<h:messages />
```

- zobrazí všechny zprávy

```
<h:inputText id="num1id" .../>
<h:message for="num1id" style="color: red" />
```

- zobrazí zprávu pro danou komponentu

Lokalizace (1)

- ve face-config.xml

```
<application>
    <locale-config>
        <default-locale>en</default-locale>
        <supported-locale>en</supported-locale>
        <supported-locale>cs</supported-locale>
        <supported-locale>cs_CZ</supported-locale>
    </locale-config>
    <message-bundle>Messages</message-bundle>
</application>
```

- ve stránce

```
<f:loadBundle basename="Messages" var="bundle" />
<h:outputText value="#{bundle.enter}" />
```

Lokalizace (2)

- src/Messages_en.properties

```
enter: Enter two numbers\:
```

- src/Messages_cs.properties

```
enter: Zadej dvě čísla\:
```

- Pozor, property file musí být v kódování Latin1, musí se použít kód (naštěstí NB začaly podporovat automatickou konverzi)

Lists and Tables

- viz příklady

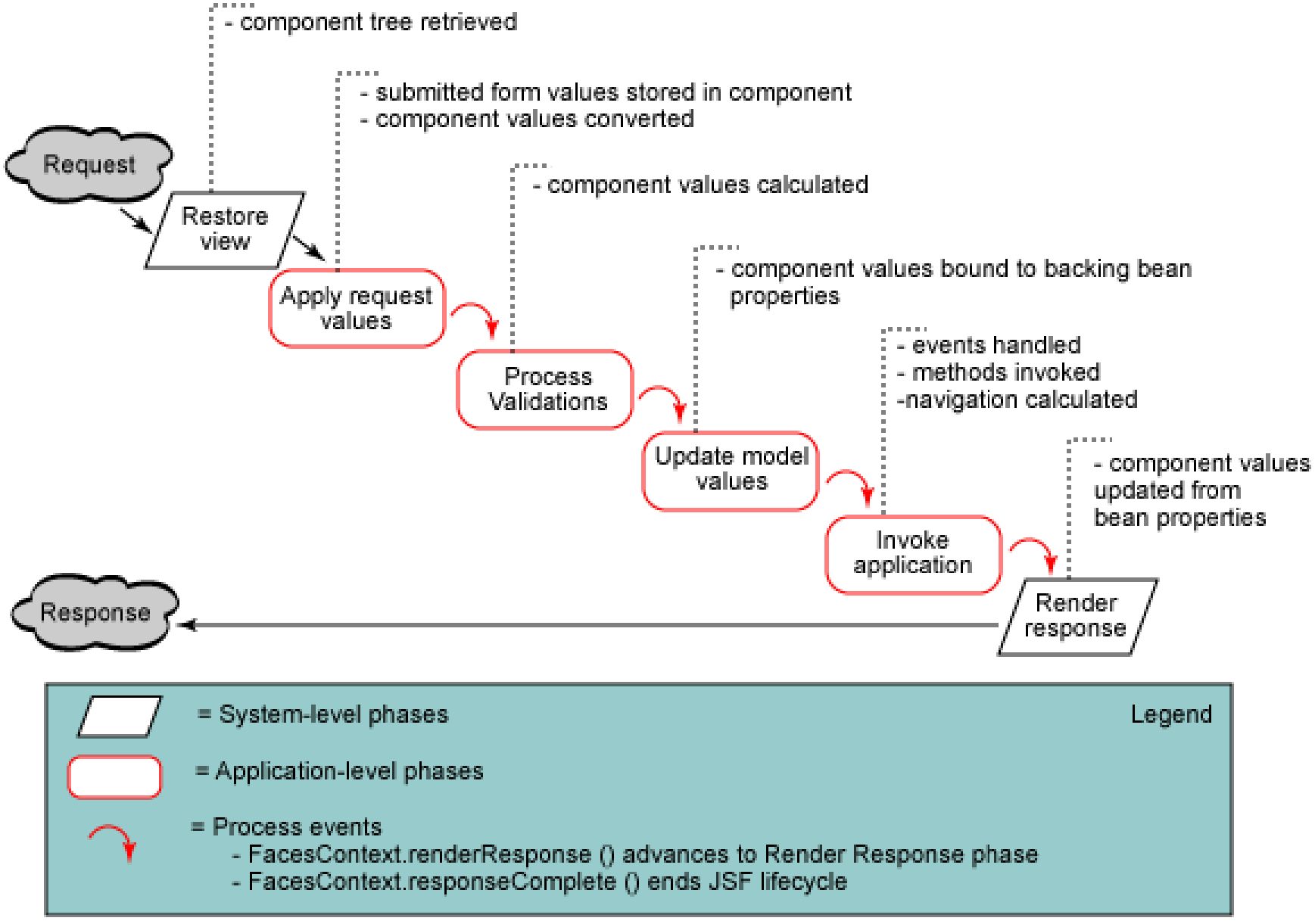
```
<h:selectOneListbox id="selectCarList"
 value="#{carBack.selectedCar}">
      <f:selectItems value="#{carBack.selectCars}" />
```

- musí dostat seznam SelectItem
- <h:dataTable id="carTable" value="#{carBack.cars}"
 var="rowCar">
- <h:column>
- <f:facet name="header">
- <h:outputText value="SPZ"/>
- </f:facet>
- <h:outputText value="#{rowCar.spz}"/>
- </h:column>
- stačí seznam a popis sloupce; jak dostanu seznam linků?

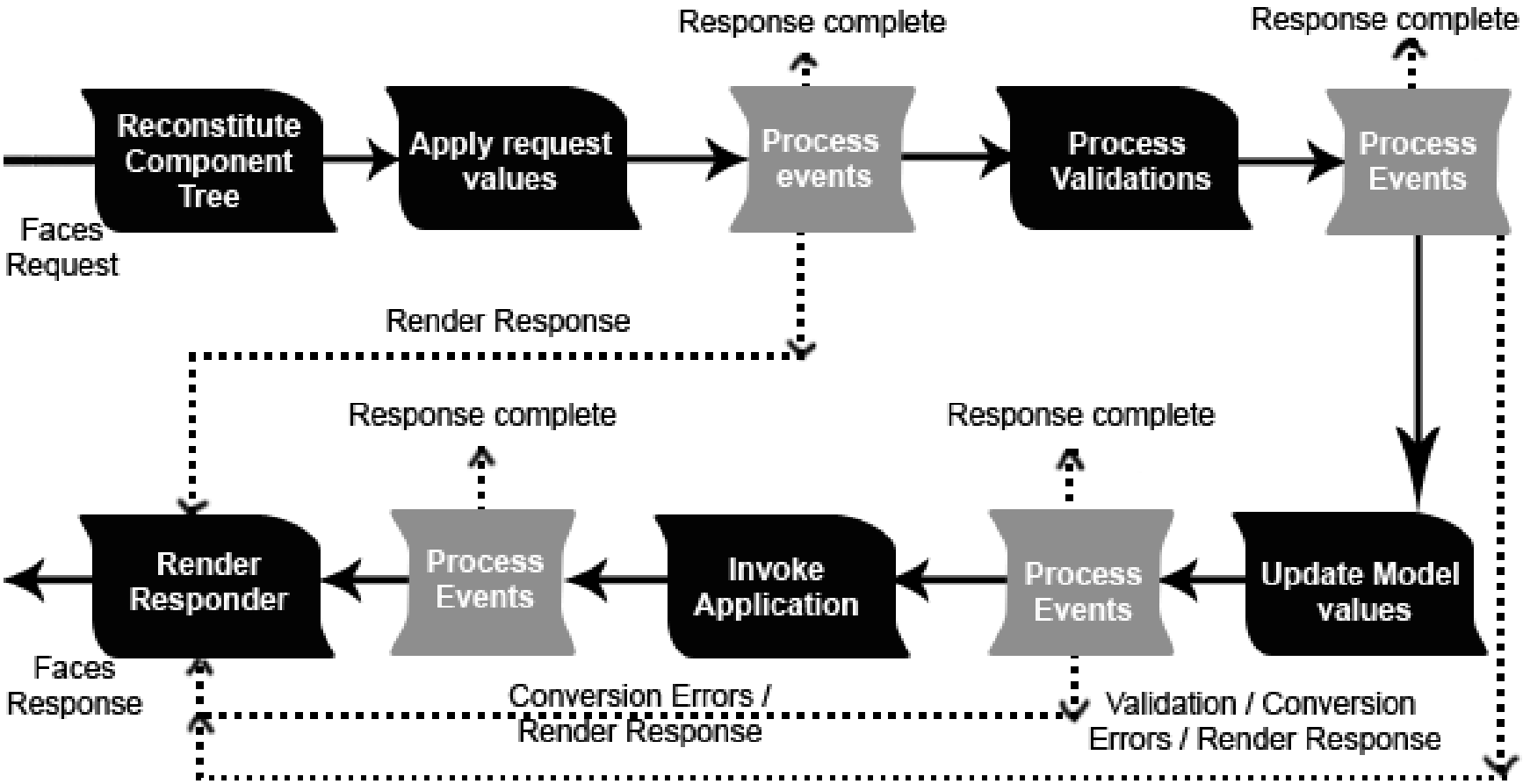
Table and Model

- abychom udělali tabulku s linky, musíme mít možnost získat informaci, ke kterému objektu se link váže. To nám zařídí model.
- Java
  - DataModel carsModel = null;
  - carsModel = new ListDataModel(getCars());
  - selectedCar = (Auto)carsModel.getRowData();
- JSF
  - <h:commandLink action="#{carBack.showCarDetail}">
  - <h:outputText value="#{rowCar.spz}"/>
  - </h:commandLink>

Lifecycle - comments

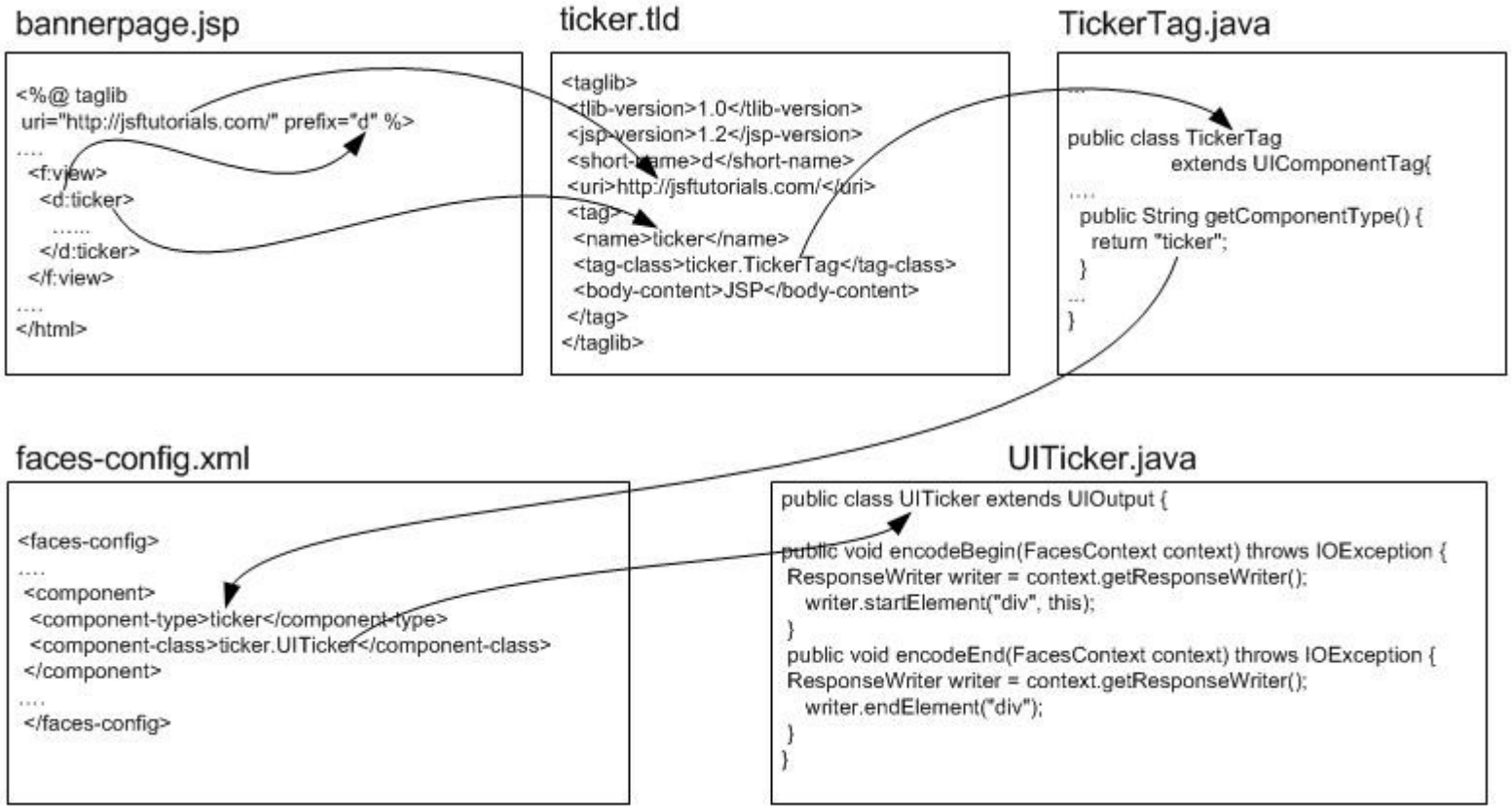


Lifecycle – exceptions



Komponenty

- provázání, definice jedné komponenty



Seznam standardních komponent

- Button
- Checkbox
- Checkbox List
- Component Label
- Data Table
- Dropdown List
- Faces Form
- Formatted Output
- Grid Panel
- Group Panel
- Hidden Field
- Hyperlink
- Image
- Inline Message
- Link Action
- Listbox
- Message List
- Multi Line Text Area
- Multi Select Listbox
- Output Text
- Radio Button List
- Secret Field
- Text Field

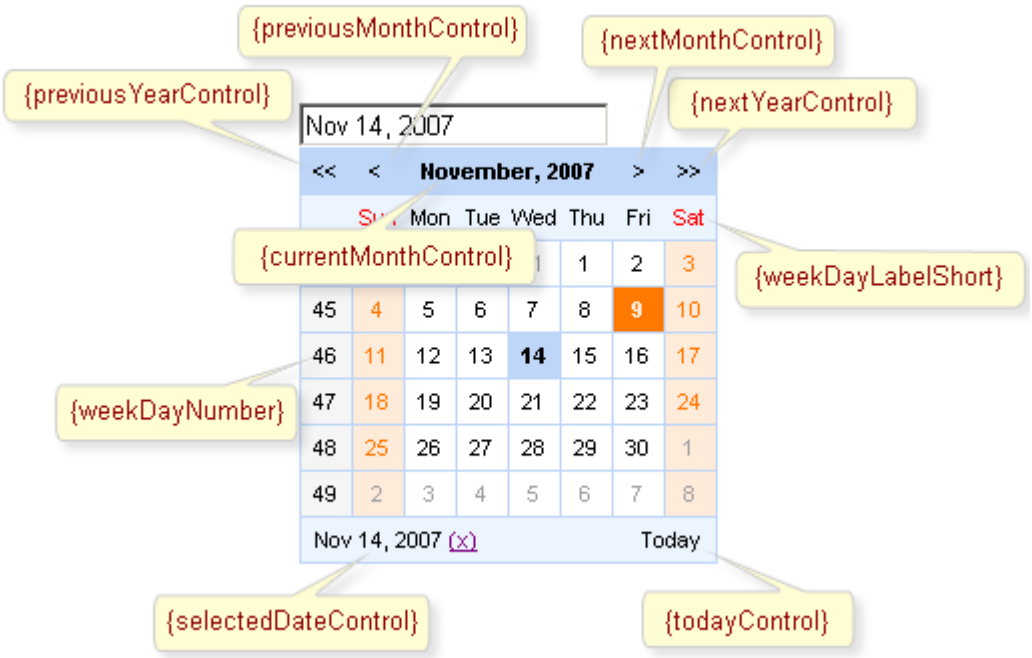
Knihovny

- Nad JSF se dají snadno budovat knihovny widgetů. Takovýchto knihoven se opět vyrojil bezpočet. Nejslavnější jsou richfaces (JBoss) a myfaces (Apache), ale vlastní má i Oracle, icefaces, ajax4jsf...
- Většinou se jedná o celé soubory UI komponent, nad kterými se dá postavit aplikace.
- Některé knihovny pouze rozšiřují stávající komponenty o AJAX chování.
- Zkuste v Googlu „jsf components“.

Richfaces

- http://www.jboss.org/file-access/default/members/jbossrichf
- podle mě nejhezčí komponenty

```
<rich:calendar id="date" value="#{bean.dateTest}">
 <a4j:support event="ondateselected" reRender="mainTable"/>
</rich:calendar>
```



MyFaces

- Mají šikovnou vlastnost, kdy normální HTML tagy jsou doplněny o JSF renderery. Designer tedy normálně navrhne stránky, kterým programátor pouze přidá atribut, čím se budou renderovat. Soubor se tedy zobrazí staticky jako normální HTML stránka, ale dynamicky ji JSF nakreslí ve své režii.

Co dál

- lepší integrace EJB a web. interface – SEAM (opět Gevin King)
  - co možná největší nahrazení backing bean anotacemi
  - v EJB 3.1 bude jako WebBeans (JSR-299)
- vyčistit chyby v návrhu – facelety
  - JSF trpí některými neduhy, které řeší nová knihovna, která je s JSF 100 % kompatibilní, pouze dělá některé akce v jiném pořadí
  - známý je problém s vytvářením komponent, kdy při prvním spuštění stránky dojde k chybě, ale při druhém je vše v pořádku.

Links

- http://java.sun.com/javaee/5/docs/tutorial/doc/bnaph.html
  - JSF v JEE tutorialu
- http://jsfcentral.com
- http://www.coreservlets.com/JSF-Tutorial/
- https://facelets.dev.java.net/nonav/docs/dev/docbook.html
- http://www.netbeans.org/kb/docs/web/quickstart-facelets-in
- http://www.icefaces.org/main/resources/tutorials.iface
- http://facesmatrix.net

Závěr

- Nejlepší cesta (když teď chápete základy)
  - projděte si nějaký tutoriál
  - zkuste si svoji vlastní aplikaci
- Hodně úspěchů ve vytváření graficky přitažlivých a funkčně bohatých aplikací!