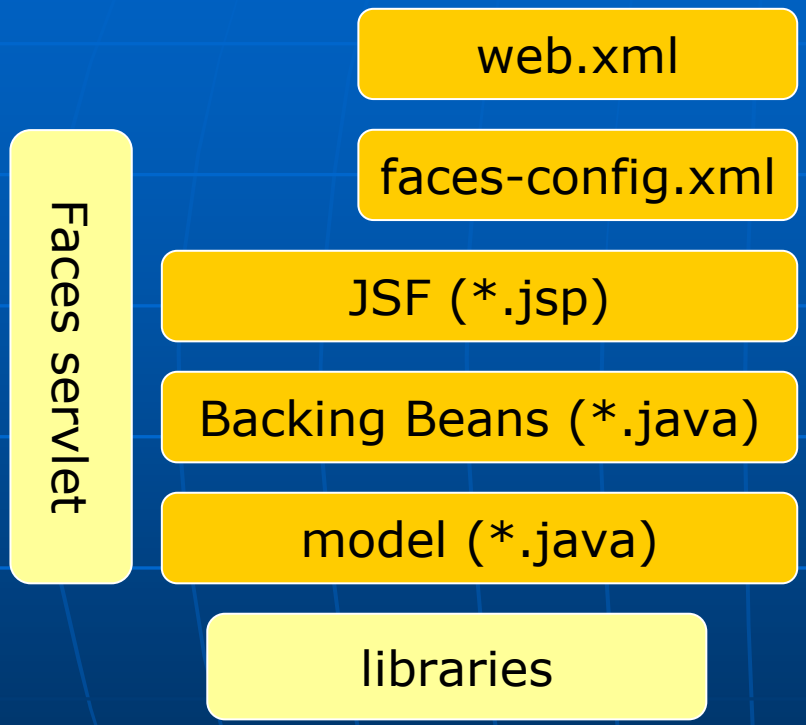
JavaServer Faces

Zdeněk Troníček

JSF aplikace

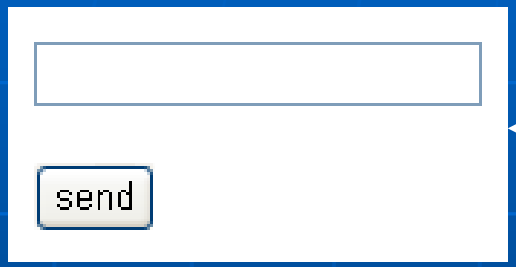



```
<%@taglib prefix="f" uri="..." %>
<%@taglib prefix="h" uri="..." %>
<html>
   <head>...</head>
   <body>
      <f:view>
         Hi,
         <h:outputText
          value="#{user.name}"/>
      </f:view>
   </body>
</html>
```

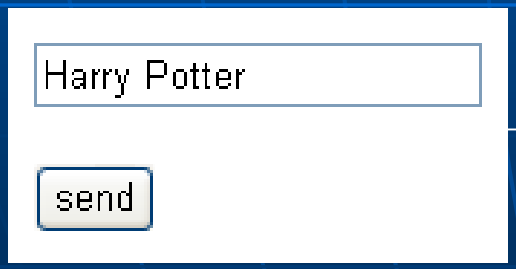
expression language: #{...}

Requests

initial request



postback request




```
<%@taglib prefix="f" uri="..." %>
<%@taglib prefix="h" uri="..." %>
<html>
   <head>...</head>
   <body>
      <f:view>
         <h:form>
            <h:inputText
             value="#{user.name}"/>
            <h:commandButton
             value="send"
             action="#{user.store}"/>
         <h:form>
      </f:view>
   </body>
</html>
```

Backing Bean

```
public class UserBean {
    private String name;
    private int age;
    //public UserBean() { }
    public String getName() {
        return user;
    }
    public void setName( String name ) {
        this.name = name;
    }
    public int getAge() {
        return age;
    }
    public void setAge( int age ) {
        this.age = age;
    }
}
```

```
<h:outputText
  value="#{user.name}"/>
```

initial response: getter
postback response: getter

```
<h:inputText
  value="#{user.age}"/>
```

initial response: getter
postback request: setter
postback response: getter

Příklad

```
<%@taglib prefix="f"
  uri="http://java.sun.com/jsf/core" %>
<%@taglib prefix="h"
  uri="http://java.sun.com/jsf/html" %>
<html>
  <head>...</head>
  <body>
    <f:view>
      <h:form>
        <h:inputText value="#{user.name}"/>
        <h:commandButton value="send"
          action="#{user.store}"/>
      </h:form>
    </f:view>
  </body>
</html>
```

Initial:
1. UserBean.getName()

Postback:
2. UserBean.setName()
3. UserBean.store()
4. UserBean.getName()

Managed Bean (faces-config.xml)

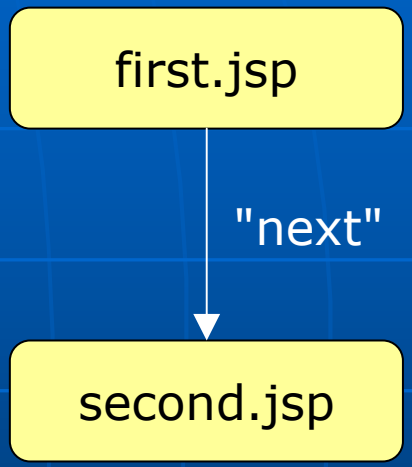
```
<faces-config>
  ...
  <managed-bean>
    <managed-bean-name>user</managed-bean-name>
    <managed-bean-class>
       x36tjv.UserBean
    </managed-bean-class>
    <managed-bean-scope>request</managed-bean-scope>
  </managed-bean>
  ...
</faces-config>
```

scope

request	application
session	none

Navigace (faces-config.xml)

```
<faces-config>
  ...
  <navigation-rule>
    <from-view-id>/first.jsp</from-view-id>
    <navigation-case>
      <from-outcome>next</from-outcome>
      <to-view-id>/second.jsp</to-view-id>
    </navigation-case>
  </navigation-rule>
  ...
</faces-config>
```



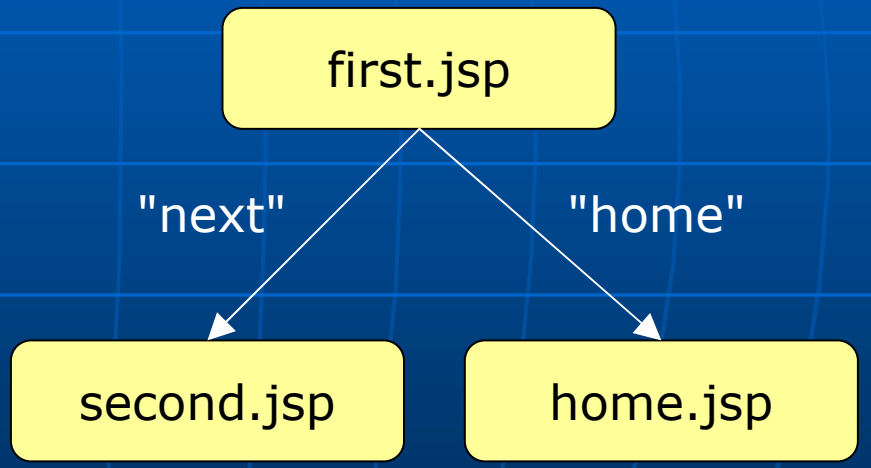
statická navigace:

```
<h:commandButton value="send" action="next"/>
```

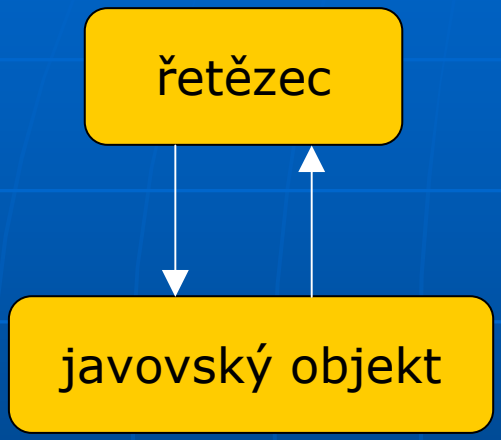
Dynamická navigace

```
<h:commandButton value="send" action="#{user.store}"/>
```

```
public class UserBean {
    ...
    public String store() {
        if (...) {
            return "home";
        }
        return "next";
    }
}
```



Konverze




```
<h:inputText value="#{dvd.year}"
   required="true">
   <f:convertDateTime pattern="yyyy"/>
</h:inputText>
```

BigDecimalConverter
BigIntegerConverter
BooleanConverter
ByteConverter
CharacterConverter
DateTimeConverter
DoubleConverter
EnumConverter
FloatConverter
IntegerConverter
LongConverter
NumberConverter
ShortConverter

Konvertor

```
class PhoneConverter implements Converter {
    public Object getAsObject( FacesContext fc,
        UIComponent comp, String value ) {
        // String → Object (ConverterException)
    }
    public String getAsString( FacesContext fc,
        UIComponent comp, Object value ) {
        // Object → String (ConverterException)
    }
}
```

faces-config.xml

```
<converter>
  <converter-for-class>x36tjv.Phone</converter-for-class>
  <converter-class>x36tjv.PhoneConverter</converter-class>
</converter>
```

Validace

form data validation
business-logic validation

```
<h:inputText value="#{app.length}">
   <f:validateDoubleRange minimum="0"
    maximum="100"/>
</h:inputText>
```

DoubleRangeValidator
LongRangeValidator
LengthValidator – délka řetězce

Validátor

```
class PhoneValidator implements Validator {
    public void validate( FacesContext fc,
        UIComponent comp, Object value )
        throws ValidatorException { ... }
}
```

faces-config.xml

```
<validator>
    <validator-id>phoneValidator</validator-id>
    <validator-class>x36tjv.PhoneValidator</validator-class>
</validator>
```

JSP

```
<h:inputText value="#{user.phone}">
    <f:validator validatorId="phoneValidator"/>
</h:inputText>
```

Lokalizace

JSP

```
<f:view>
    <f:loadBundle basename="messages"
      var="msg"/>
    <h:outputText
      value="#{msg.appName}"/>
</f:view>
```

messages_en.properties

```
appName=DVD Library
```

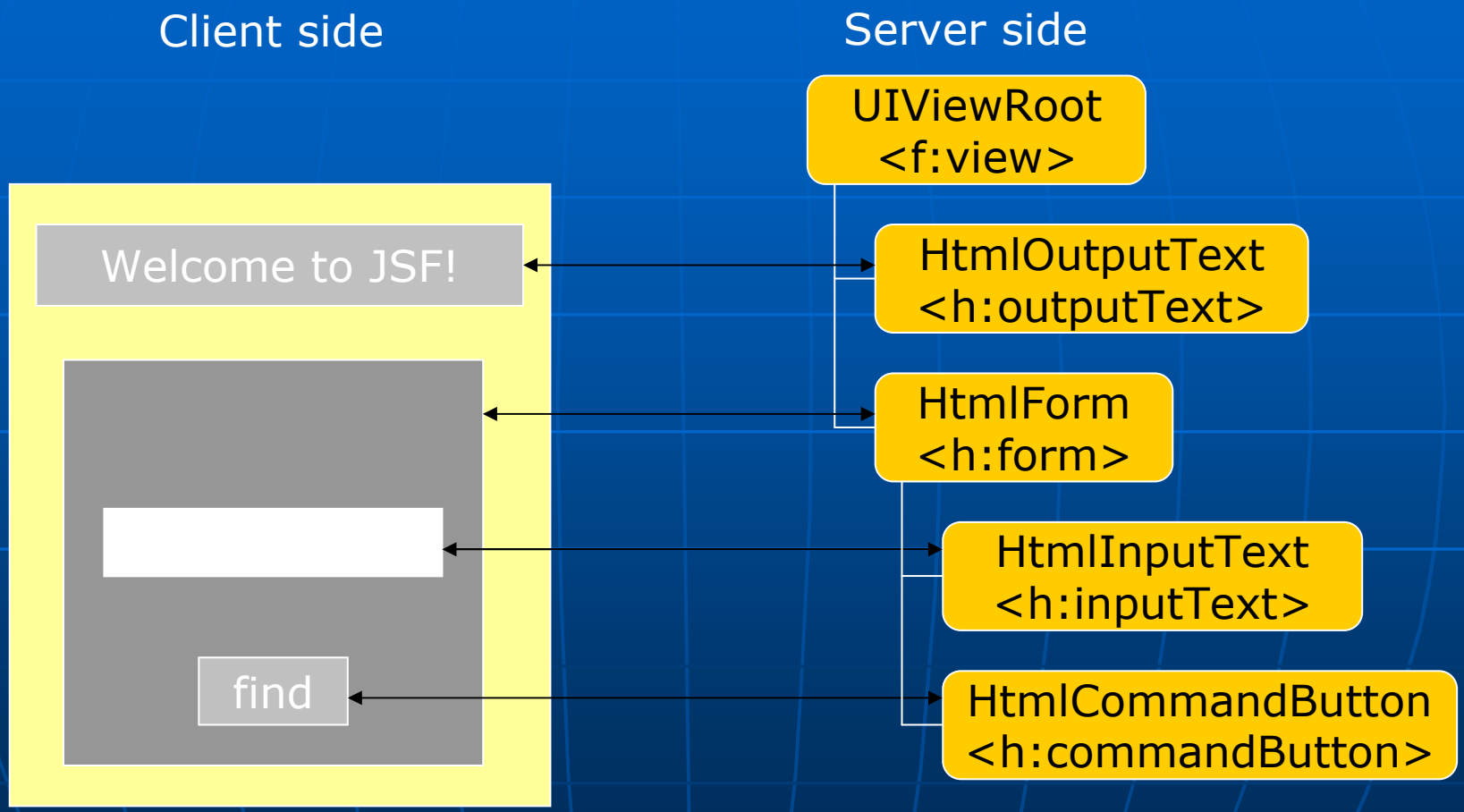
messages_cs.properties

```
appName=Knihovna DVD
```

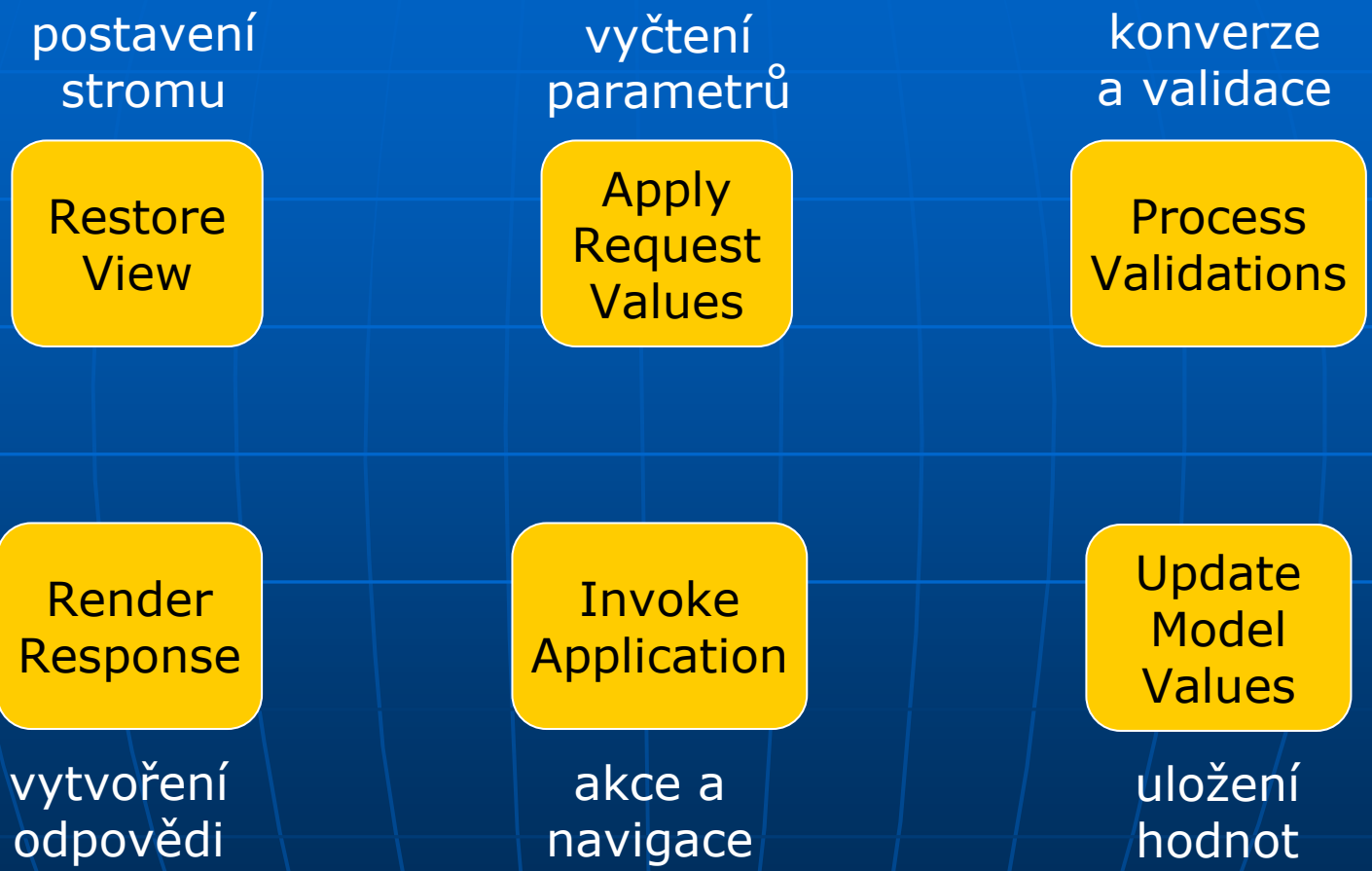
faces-config.xml

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

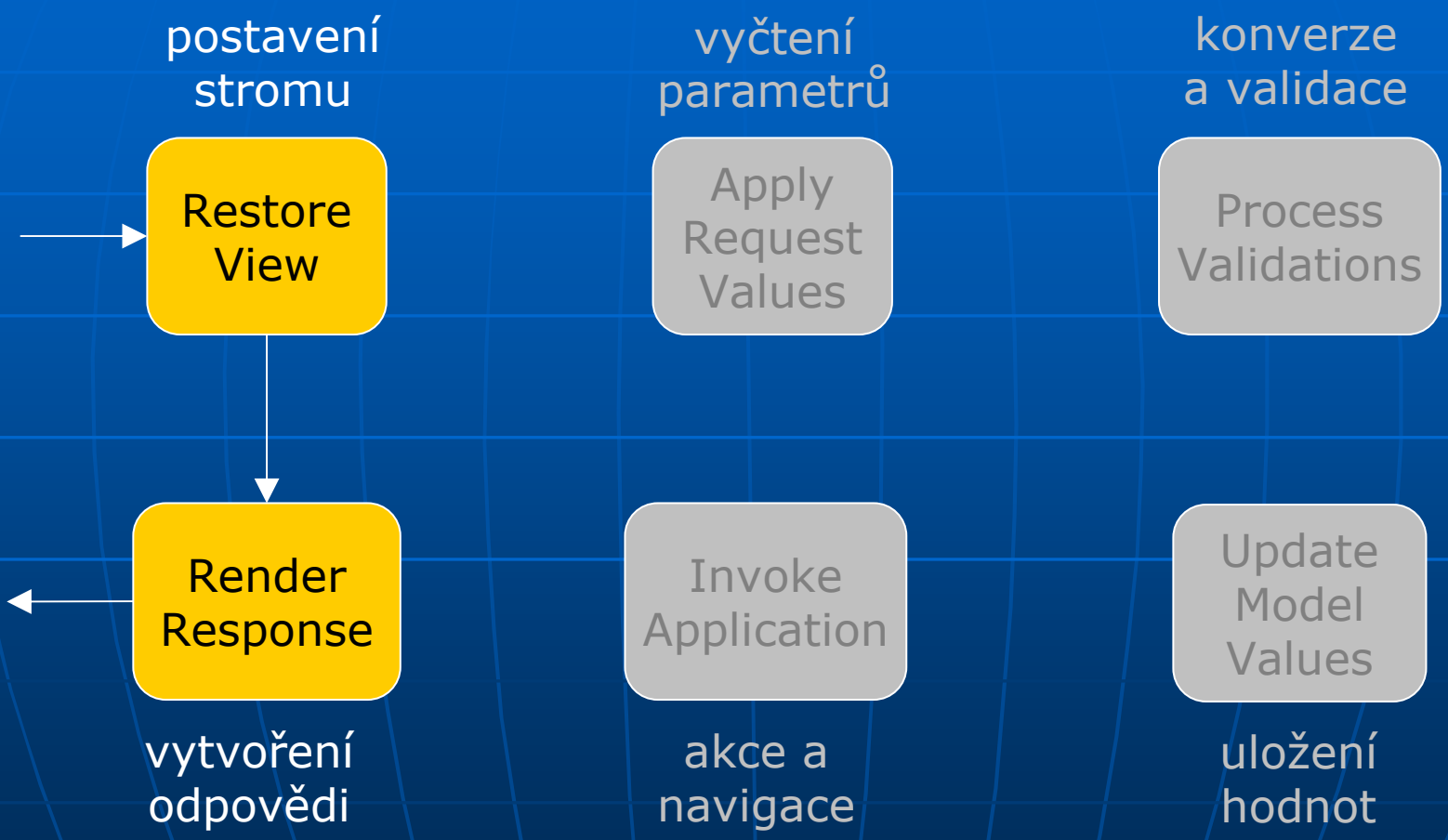
Component Tree



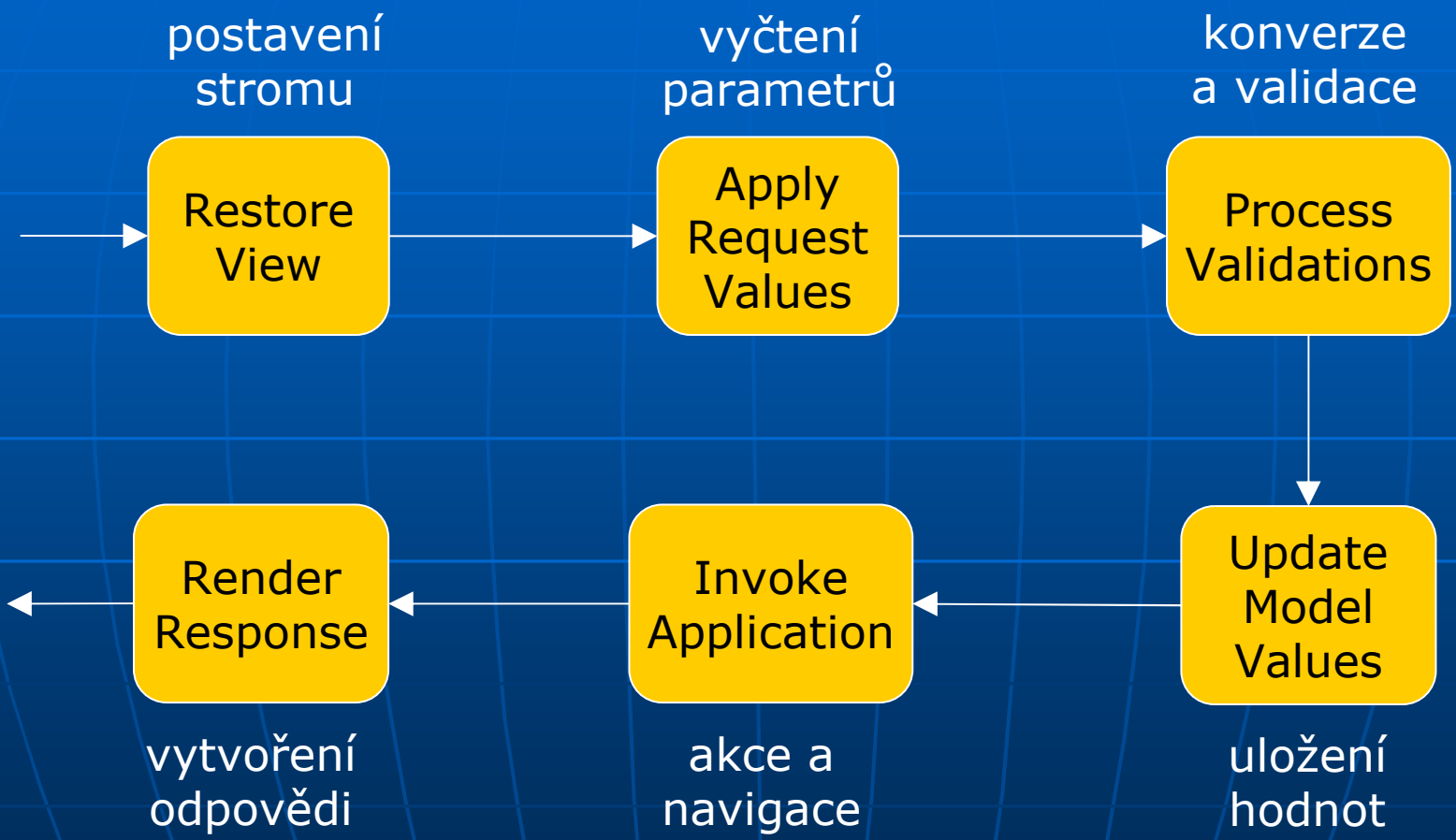
Request Life Cycle



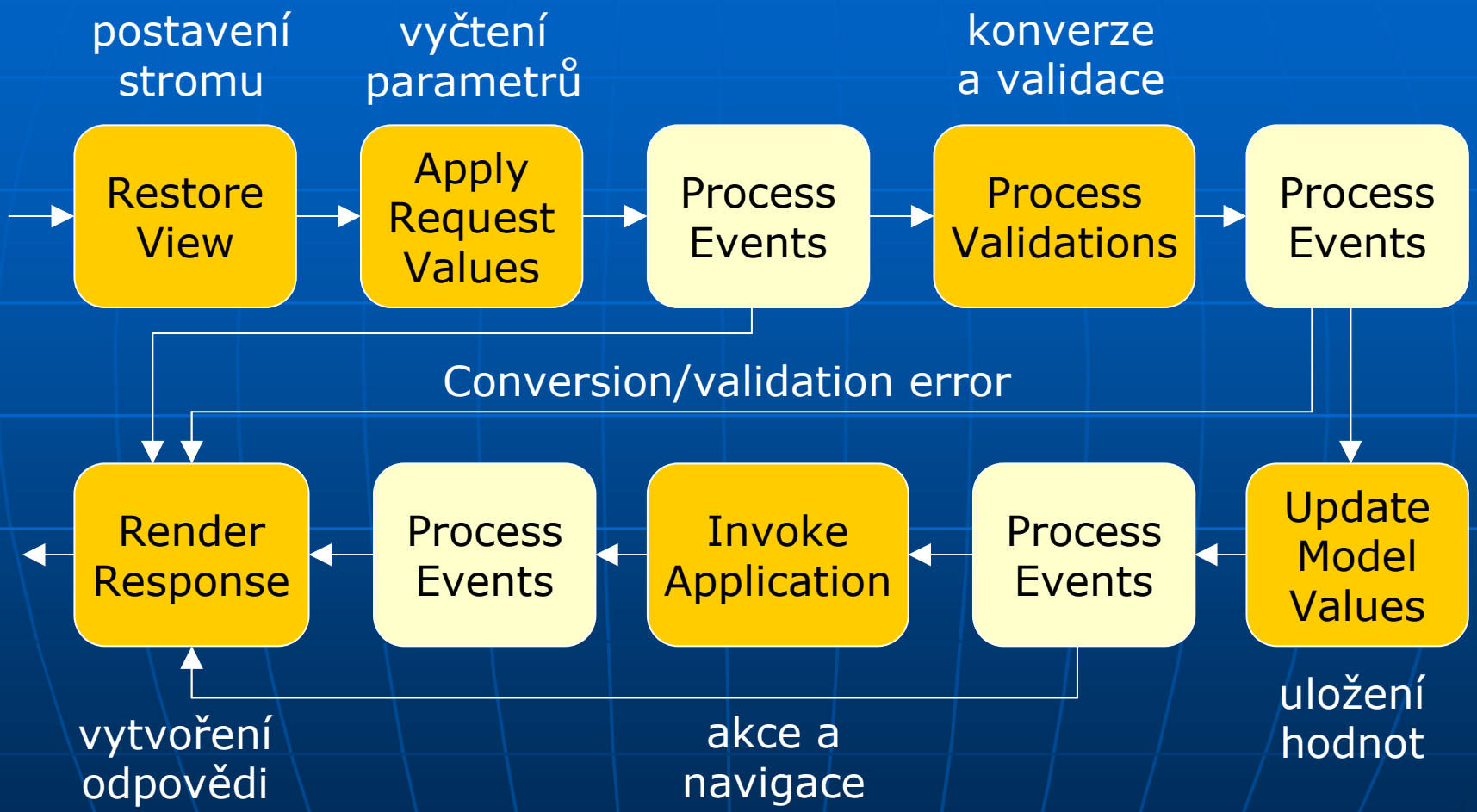
Initial Request



Postback Request



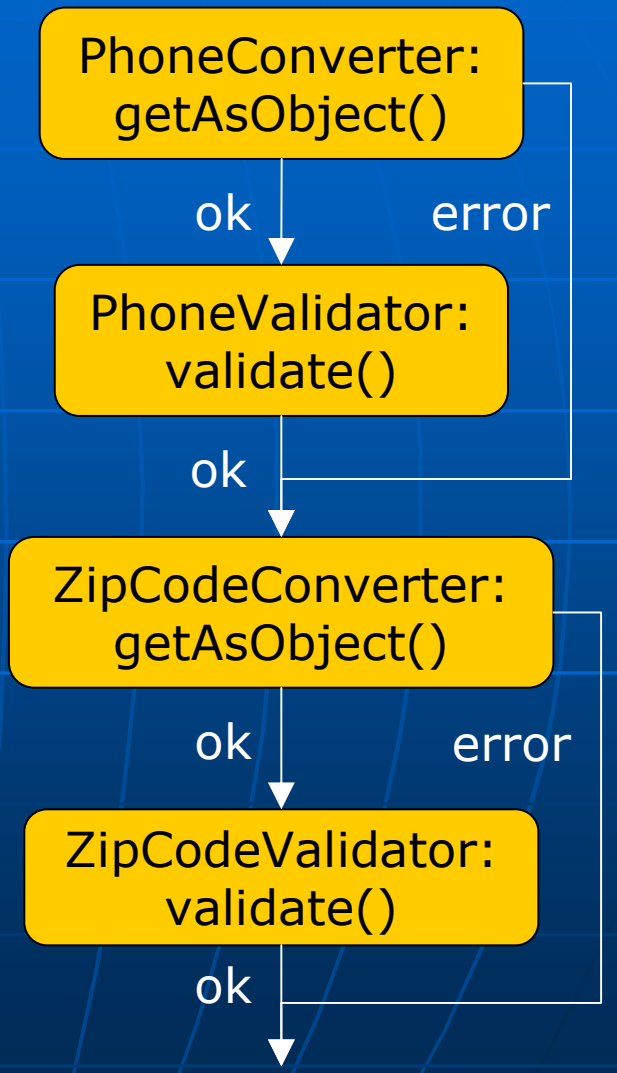
Events


postavení stromu
vyčtení parametrů
konverze a validace
Restore View
Apply Request Values
Process Events
Process Validations
Process Events
Conversion/validation error
Render Response
Process Events
Invoke Application
Process Events
Update Model Values
vytvoření odpovědi
akce a navigace
uložení hodnot

Process Validations

```
<h:inputText value="#{user.phone}">
  <f:converter
    converterId="PhoneConverter"/>
  <f:validator
    validatorId="PhoneValidator"/>
</h:inputText>
<h:inputText value="#{addr.zipCode}">
  <f:converter
    converterId="ZipCodeConverter"/>
  <f:validator
    validatorId="ZipCodeValidator"/>
</h:inputText>
<h:messages/>
```

FacesContext: addMessage()



Render Response

Zahrnuje:
- vytvoření odpovědi
- uložení stavu komponent

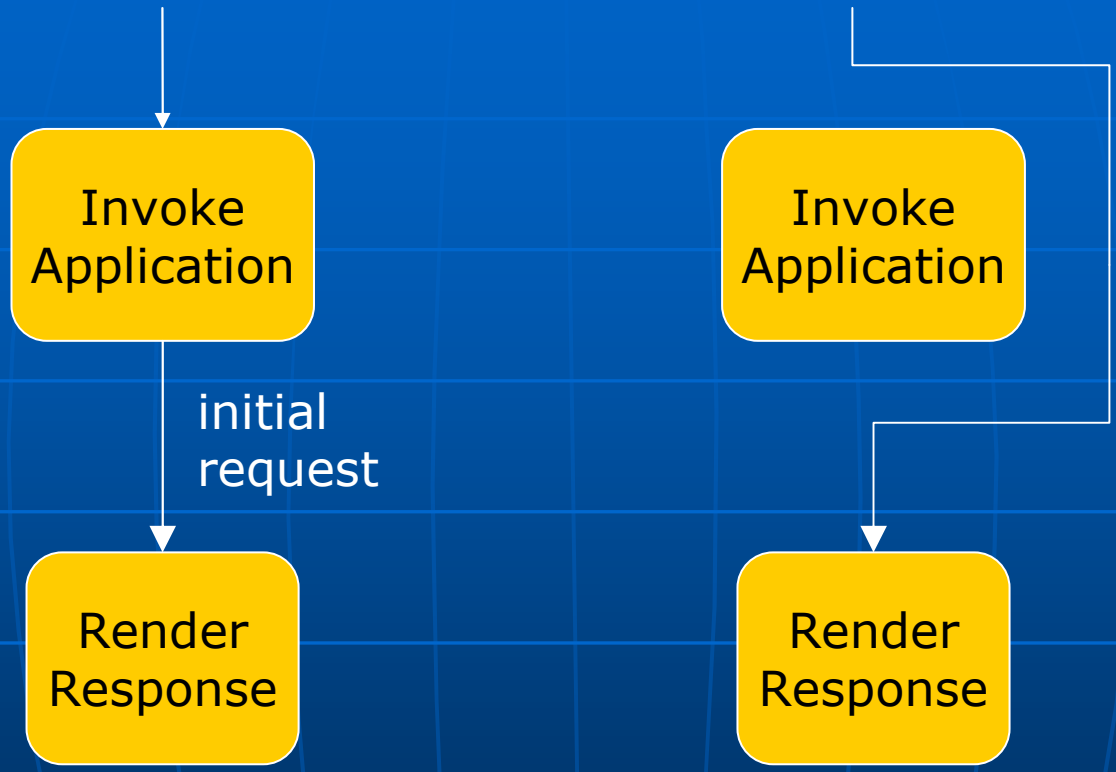
StateHolder

```
Object saveState( FacesContext fc )
void restoreState( FacesContext fc,
  Object state )
```

web.xml

```
<context-param>
    <param-name>
      javax.faces.STATE_SAVING_METHOD
    </param-name>
    <param-value>client</param-value>
</context-param>
```

Render Response (2)



navigace → nový strom komponent

bez navigace → původní strom komponent

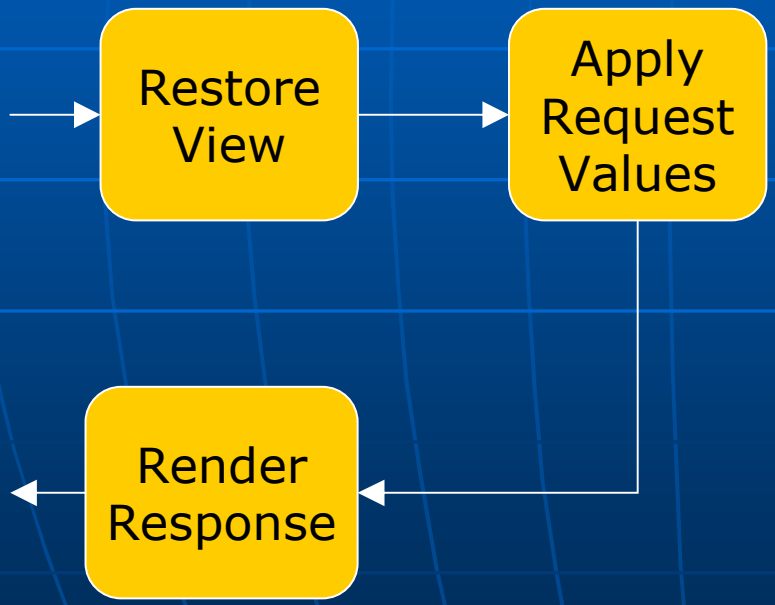
Property “immediate”

```
<h:commandLink immediate="true" … />
<h:commandButton immediate="true" … />
```

ActionSource

ActionEvent

ActionListener



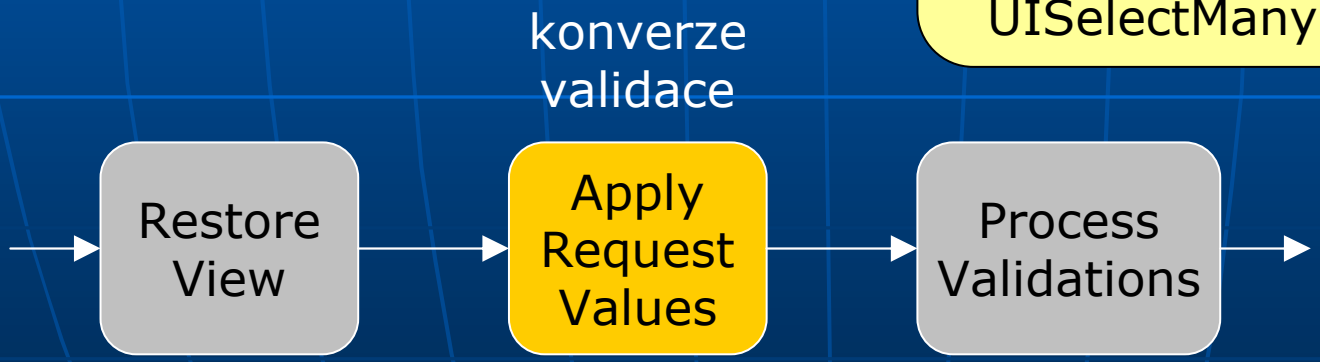
DefaultActionListener

1. akce
2. navigace

Property “immediate” (2)

```
<h:inputText immediate="true" … />
<h:selectBooleanCheckbox immediate="true" … />
```

EditableValueHolder:
- UIInput
- UISelectBoolean
- UISelectOne
- UISelectMany



JSF komponenty

- Tag

  inputText, selectOneListbox, selectOneMenu, selectOneRadio, selectManyListbox,...

- UIComponent

  UIInput, UISelectOne, UISelectMany,...

- Renderer

  Text, Secret, Listbox, Menu, Radio,...

Př.: UIInput + Text = inputText
UISelectOne + Listbox = selectOneListbox

Q&A

Děkuji za pozornost