class: inverse, center, middle

# JavaServer Faces

class: inverse, center, middle

# Tematika

## Tematika

* Egyszerű oldal felépítése
* Kérés életútja, navigáció, átirányítás és hibakezelés
* Képernyő fejlesztés menete (megjelenítés, űrlap)
* Navigáció
* Konvertálás, validáció
* I18N
* Stílusok
* Facelets alapú sablonkezelés
* PrimeFaces
* Összetett és egyedi komponensfejlesztés

## Források

* Antonio Goncalves: Beginning Java EE 7 (Expert Voice in Java)
* David Geary,‎ Cay S. Horstmann: Core JavaServer Faces (3rd Edition)
* [Java Platform, Enterprise Edition 7: The Java EE Tutorial](https://docs.oracle.com/javaee/7/tutorial/index.html)

class: inverse, center, middle

## JSF

## JSF koncepció

* Java EE specifikáció webes felhasználói felület implementálására
* Komponensek, események, Java beanek kérés/válasz helyett
* Model-View-Controller (MVC) tervezési minta
* Könnyen tanulható, egységes, szabványos

## JSF implementációk

* Specifikáció
* [JSR 372](https://jcp.org/en/jsr/detail?id=372): JavaServer Faces (JSF 2.3) Specification
* [JSF 2.3 API](https://javaserverfaces.github.io/docs/2.3/index.html)
* Referencia implementáció: [Mojarra](https://javaee.github.io/javaserverfaces-spec/)
* További implementáció: [Apache MyFaces](http://myfaces.apache.org/)

## JSF verziók

* 1.2 - Java EE 5
* 2.0 - Java EE 6
* 2.2 - Java EE 7 (2013. június)
* CDI 1.1
* Bean Validation 1.1
* 2.3 - Java EE 8 (2017. szeptember)
* CDI 2.0
* Servlet 4.0
* Bean Validation 2.0

## JSF 2.0 újdonságok

* Egyszerűbb navigáció
* Egyszerűbb konfiguráció annotációkkal
* Egyszerűbb komponensfejlesztés
* POST kérések helyett GET kérések (könyvjelzőzhetőség)
* Erőforrások kezelése
* Beépített AJAX támogatás (jsf.js és <f:ajax> tag)

## JSF 2.2 újdonságok

* HTML tagek használata
* Faces Flow - képernyők egymás utánisága
* AJAX kérések sorrendbe állítása, egy időn belül csak az utolsó elküldése
* Injection több helyre (converter, validator, komponens)
* JSF 2.0 saját scoping és bean mechanizmusa helyett a standard CDI (javax.faces.bean csomag deprecated)

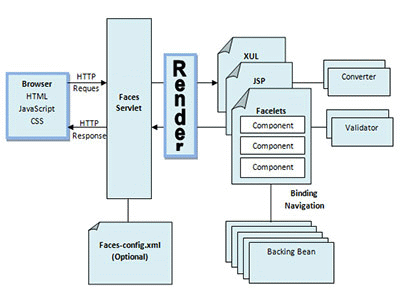
## JSF 2.3 újdonságok

* Injektálható és EL-ben használható implicit JSF objektumok
* WebSocket integráció
* AJAX hívás <h:commandScript> komponenssel
* Osztályszintű bean validation
* Java 8 java.time támogatás
* Component Search Expression framework

## Környezet

* Jetty, Tomcat - a JSF a CDI-re épül, a web konténerek nem tartalmazzák
* Körülményesen integrálható
* Java EE 8 alkalmazásszerver
* Spring Frameworkkel integrálható

## JSF architektúra áttekintés



JSF architektúra

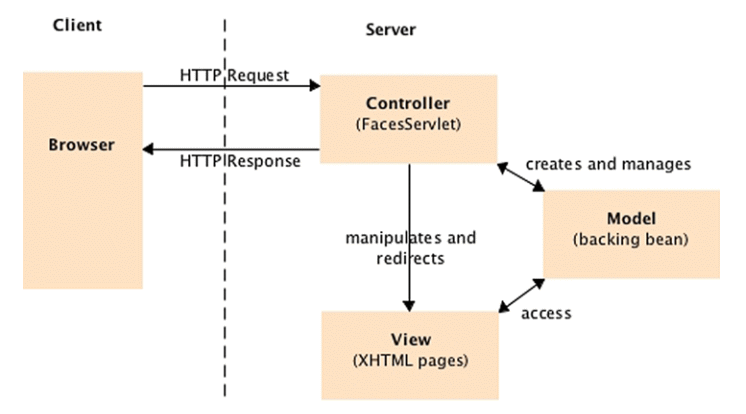
## JSF fogalmai

* Faces Servlet (Front Controller tervezési minta)
* Opcionális konfiguráció a faces-config.xml fájlban, helyette annotációk
* Oldalak és komponensek: több page declaration language (PDL) vagy view declaration language (VDL), JSF 2.0 óta Facelets javasolt
* Expression Language: komponens és backing bean kapcsolata (variable és action bind)
* Renderer: komponens megjelenítése, felhasználói adatbevitel
* Backing beans: üzleti logika és navigáció

## JSF fogalmai

* Converter: szöveg és Java adattípusok közötti konverziók
* Validator: adatbevitel ellenőrzése, delegáció Bean Validation felé

## MVC



MVC

## Egyszerű Java EE alkalmazás - pom.xml

A pom.xml állományban Java EE függőség:

<dependency>  
 <groupId>javax</groupId>  
 <artifactId>javaee-api</artifactId>  
 <version>8.0</version>  
 <scope>provided</scope>  
</dependency>

## Egyszerű Java EE alkalmazás - faces-config.xml

<?xml version="1.0" encoding="UTF-8"?>  
<faces-config  
 xmlns="http://xmlns.jcp.org/xml/ns/javaee"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee/web-facesconfig\_2\_3.xsd"  
 version="2.3">  
</faces-config>

Ekkor nem szükséges web.xml!

## Egyszerű Java EE alkalmazás - Facelets

<?xml version="1.0" encoding="UTF-8"?>  
<!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:ui="http://xmlns.jcp.org/jsf/facelets">  
  
<h:head>  
 <title>JSF Hello World</title>  
</h:head>  
<h:body>  
 <h1>JSF Hello World</h1>  
  
 #{helloController.message}  
</h:body>  
</html>

## Egyszerű Java EE alkalmazás - Backing bean

package hello;  
  
import javax.enterprise.context.RequestScoped;  
import javax.inject.Named;  
  
@Named  
@RequestScoped  
public class HelloController {  
  
 public String getMessage() {  
 return "Hello JSF!";  
 }  
}

## Deploy

* Deploy után elérhető a /faces/index.xhtml címen

class: inverse, center, middle

## Spring Framework integráció

## Spring Framework integráció

* [JoinFaces](http://joinfaces.org/) - JSF Spring Boot Starters

## pom.xml

<dependencyManagement>  
 <dependencies>  
 <dependency>  
 <groupId>org.joinfaces</groupId>  
 <artifactId>joinfaces-dependencies</artifactId>  
 <version>4.0.4</version>  
 <type>pom</type>  
 <scope>import</scope>  
 </dependency>  
 </dependencies>  
</dependencyManagement>  
  
<dependencies>  
   
 <!-- ... -->  
   
 <dependency>  
 <groupId>org.joinfaces</groupId>  
 <artifactId>jsf-spring-boot-starter</artifactId>  
 </dependency>  
</dependencies>

## Facelets

<?xml version="1.0" encoding="UTF-8"?>  
<!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:ui="http://xmlns.jcp.org/jsf/facelets">  
  
<h:head>  
 <title>JSF Hello World</title>  
</h:head>  
<h:body>  
 <h1>JSF Hello World</h1>  
  
 #{helloController.message}  
</h:body>  
</html>

## Backing bean

package hello;  
  
import org.springframework.stereotype.Component;  
import org.springframework.web.context.annotation.RequestScope;  
  
@Component  
@RequestScope  
public class HelloController {  
  
 public String getMessage() {  
 return "Hello JSF!";  
 }  
}

## Deploy

* Nem szükséges web.xml
* Nem szükséges faces-config.xml

class: inverse, center, middle

## Űrlap kezelés

# Redirect after post

* Űrlap POST után átirányítás
* History működése
* Refresh működése
* Könyvjelzőzhetőség

# Backing bean

* Metódus, String visszatérési értékkel
* Navigációs szabályok
* Konkrét oldal neve
* null esetén az oldalon marad
* Átirányítás: ?faces-redirect=true

# Oldal

* h:form
* Beviteli komponensek
* Pl. h:inputText, value attribútum: backing bean bind
* h:commandButton
* value attribútum: szöveg
* action attribútum: backing bean metódus, paraméterezhető, nem EL esetén oldal neve
* h:commandLink

## Form példa

<h:form>  
 <h:inputText id="nameInput" value="#{addLocationController.location.name}"></h:inputText>  
 <h:commandButton value="Add" action="#{addLocationController.addLocation()}"></h:commandButton>  
</h:form>

## Input elemek

* h:inputHidden
* h:inputSecret
* h:inputText
* h:inputTextarea
* h:inputFile

## Select elemek

* h:selectBooleanCheckbox
* h:selectManyCheckbox
* h:selectManyListbox
* h:selectManyMenu
* h:selectOneListbox
* h:selectOneMenu
* h:selectOneRadio

Belül f:selectItem vagy f:selectItems tagek

# Flash

FacesContext.getCurrentInstance().getExternalContext()  
 .getFlash().put("successMessage", "location\_has\_created");

<c:if test="#{not empty flash['successMessage']}">  
 #{msgs[flash['successMessage']]}  
</c:if>

## Konvertálás

* Típus egyeztetés
* h:message, for attribútummal
* h:messages tag

class: inverse, center, middle

## Stílus

## Erőforrás kezelés

* resources könyvtárból vagy jar META-INF/resources könyvtárából

<h:outputStylesheet library="css" name="bootstrap.min.css" />  
<h:outputScript library="js" name="bootstrap.min.js" />

class: inverse, center, middle

## Konvertálás és validálás

# Konverzió

* Beépített converterek, f:convertDateTime, f:convertNumber

<h:outputText value="#{location.date}">  
 <f:convertDateTime pattern="yyyy-MM-dd" />  
</h:outputText>

<h:inputText id="dateInput" value="#{addLocationController.location.date}">  
 <f:convertDateTime pattern="yyyy-MM-dd" />  
</h:inputText>

# Converter fejlesztése

@FacesConverter("localDateConverter")  
public class LocalDateConverter implements Converter{  
  
 private static final DateTimeFormatter FORMATTER   
 = DateTimeFormatter.ofPattern("yyyy.MM.dd.");  
  
 @Override  
 public Object getAsObject(FacesContext facesContext,   
 UIComponent uiComponent, String s) {  
 return LocalDate.parse(s, FORMATTER);  
 }  
  
 @Override  
 public String getAsString(FacesContext facesContext,   
 UIComponent uiComponent, Object o) {  
 return ((LocalDate) o).format(FORMATTER);  
 }  
}

## Konverter típushoz

@FacesConverter(forClass=Phone.class)

# Konverter használata

<h:outputText value="#{localDateHelloController.date}">  
 <f:converter converterId="localDateConverter" />  
</h:outputText>

## Paraméterezhető konverterek

https://stackoverflow.com/a/31201621

# Required validátor

* required attribútum

<h:inputText id="nameInput" value="#{addLocationController.location.name}" required="true"></h:inputText>

# Beépített validátorok

* DoubleRangeValidator
* LengthValidator
* LongRangeValidator
* MethodExpressionValidator
* RequiredValidator
* RegexValidator

<h:inputText id="nameInput" value="#{addLocationController.location.name}">  
 <f:validateLength minimum="2" maximum="20"/>  
</h:inputText>

## Saját validátor

@FacesValidator("nameValidator")  
public class NameValidator implements Validator {  
  
 @Override  
 public void validate(FacesContext facesContext, UIComponent uiComponent, Object o) throws ValidatorException {  
   
 }  
}

<h:inputText id="nameInput" value="#{addLocationController.location.name}" class="form-control">  
 <f:validator validatorId="nameValidator" />  
</h:inputText>

# Validátor paraméterezése

* Validátor nem paraméterezhető
* Komponensnek lehet attribútuma

<h:inputText id="nameInput" value="#{addLocationController.location.name}" class="form-control">  
 <f:attribute name="validatorMinLength" value="2" />  
 <f:validator validatorId="nameValidator" />  
</h:inputText>

int minLength = Integer.parseInt((String) uiComponent.getAttributes().get("validatorMinLength"));

# Bean Validation

* Out of the box működik
* f:validateBean validation group definiálásához

# Üzenetek

* FacesContext.addMessage(FacesMessage)
* Controllerben return null

FacesContext.getCurrentInstance().addMessage("locationForm:nameInput",  
 new FacesMessage(FacesMessage.SEVERITY\_ERROR, msg, msg));

class: inverse, center, middle

## I18N

## Nyelvek megadása

* Nyelvek a faces-config.xml állományban

<application>  
 <locale-config>  
 <default-locale>en</default-locale>  
 <supported-locale>hu</supported-locale>  
 </locale-config>  
</application>

## Globális bundle-ök

* Globális üzenetek a faces-config.xml állományban

<application>  
 <resource-bundle>  
 <base-name>Messages</base-name>  
 <var>msgs</var>  
 </resource-bundle>  
</application>

* Messages.properties állomány, illetve a megfelelő Resource Boundle-ök a classpath-on
* Oldalon belül: #{msgs.title}

## Lokális bundle-ök

<f:loadBundle basename="Welcome" var="localmsgs" />

Keresett a Welcome.properties

# Beépített hibaüzenetek nyelvesítése

* https://github.com/javaserverfaces/mojarra/blob/master/impl/src/main/resources/javax/faces/Messages\_en.properties
* requiredMessage attribútum
* converterMessage attribútum
* validatorMessage attribútum

## Nyelv kiírása

#{msgs.title}  
  
#{msgs['title']}  
  
<h:outputFormat value="#{msg.welcome}">  
 <f:param value="#{userController.user.name}" />  
</h:outputFormat>

# Locale beállítása

* Külső forrásból beállítható az oldalra
* Session scoped managed bean (pl. bejelentkezett felhasználó, vagy választott nyelv)
* <f:view locale="#{localeController.locale}">
* Request
* FacesContext.getCurrentInstance().getExternalContext().getRequestLocale()

## Validátor üzenet nyelvesítése

ResourceBundle bundle = facesContext.getApplication().evaluateExpressionGet(facesContext, "#{msgs}", ResourceBundle.class);  
String msg = bundle.getString("validation.location.name.error");

# Bean Validation nyelvesítése

* ValidationMessages.properties classpath-on

class: inverse, center, middle

## Template kezelés

# Template kezelés

* Template-ben placeholder definiálása: <ui:insert name="title">
* Konkrét oldalon behelyettesítés <ui:define name="title">Create a new book</ui:define>
* Include: ui:include
* Hivatkozás, relatív módon, vagy abszolút módon (/WEB-INF/templates/layout.xhtml)

class: inverse, center, middle

## AJAX

class: inverse, center, middle

## Kompozit és saját komponensek

# Saját komponens - kompozit komponens

* <composite:interface>
* <composite:attribute>
* <composite:implementation>
* cc implicit objektum attrs mezőjével hivatkozhatóak

# Saját komponens

* Megfelelő komponens osztály kiterjesztése, és implementálása
* faces-config.xml fájlban deklarálni, majd tag handler és renderer definiálása

class: inverse, center, middle

## Fájl fel és letöltés

class: inverse, center, middle

## Eseménykezelés

class: inverse, center, middle

## 3rd party libraries

## 3rd party libraries

* Leggyakoribb: PrimeFaces, RichFaces, és IceFaces