JavaServer Faces (JSF)

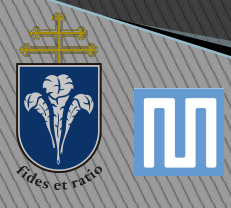
JSF technológia

Szerver oldali komponens keretrendszer, Java alapú web alkalmazások építésére:

- API, amely a következőket tartalmazza:
  - web oldal komponens modell és állapotuk kezelése
  - eseménykezelés
  - adat validáció és konverzió
  - navigáció oldalak között
- Tag könyvtárak web oldal komponensek építéséhez
- **Komponensek és szerver oldali objektumok összekapcsolása**
- CDI támogatása
- AJAX

JSF webalkalmazás építőelemei

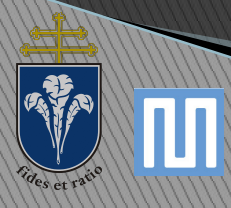
- **Web oldalak**
  - Xhtml oldalak, amelyek komponenseket tartalmaznak
  - Tagek, amelyek segítségével komponenseket adhatunk az oldalhoz
- **Beanek** – server oldali komponenesek, adatok tárolása, interceptorok, injektálás (CDI beanek, JSF beanek)
- **Telepítés leíró file** (deployment descriptor): **web.xml**
- **Applikáció konfigurációs file**: **faces-config.xml** (saját komponensek, resource bundle-k, navigációs szabályok regisztrálása)
- Felhasználó által definiált objektumok, validátorok, konverterek
- Felhasználó által definiált tagek



Facelets

Facelets egy oldalleíró nyelv, a HTML-hez hasonló szintaxissal, amelynek segítségével JSF nézetek definiálhatóak.

- Facelet, JSF és JSTL tag könyvtárak támogatása
- Expression Language (EL):
  - adatmezők elérése(setter, getter) #{personBean.person.id}
  - fv hívások pl. #{personBean.save}
  - kifejezések kiértékelése pl. #{personBean.person != null}
- Újrafelhasználhatóság támogatása: Templatek létrehozhatóak komponensekhez és oldalakhoz
- Xhtml alapú oldalak, xml tagekkel



JSF tag libraries

Könyvtár neve	Namespace	Használt prefix	Példa
Core	http://java.sun.com/jsf/core	f:	f:validateRequired
Html	http://java.sun.com/jsf/html	h:	h:inputText
Facelets	http://java.sun.com/jsf/facelets	ui:	ui:composition
Composite components	http://java.sun.com/jsf/composite	composite:	
JSTL core	http://java.sun.com/jsp/jstl/core	c:	c:if

JSF tag libraries

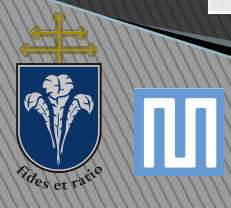
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html
      PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:fc="http://java.sun.com/jsf/composite/components"
      xmlns:f="http://java.sun">
<head>
    <title>F
</head>
<body>

<h:form>
    <h2>
        <h:o
    </h2>
```

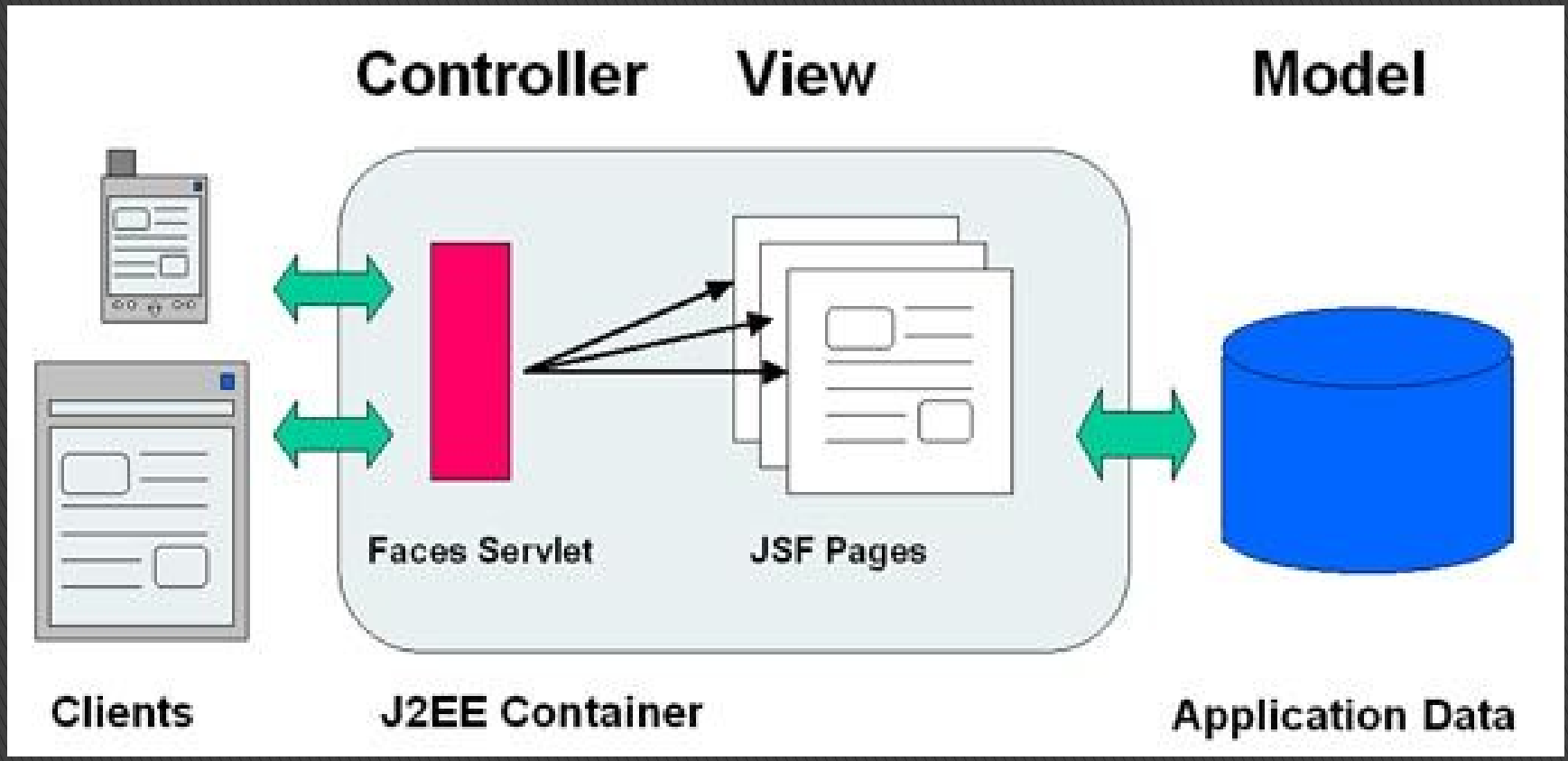
```
http://java.sun.com/jsf/composite
http://java.sun.com/jsf/composite/components
http://java.sun.com/jsf/core
http://java.sun.com/jsf/facelets
http://java.sun.com/jsf/html
http://java.sun.com/jsp/jstl/core
http://java.sun.com/jsp/jstl/functions
```

JSF html tag library

JSF component	Description
`h:form`	Inserts an XHTML `form` element into a page.
`h:commandButton`	Displays a button that triggers an event when clicked. Typically, such a button is used to submit a form's user input to the server for processing.
`h:graphicImage`	Displays an image (e.g., GIF and JPG).
`h:inputText`	Displays a text box in which the user can enter input.
`h:outputLink`	Displays a hyperlink.
`h:panelGrid`	Displays an XHTML `table` element.
`h:selectOneMenu`	Displays a drop-down list of choices from which the user can make a selection.
`h:selectOneRadio`	Displays a set of radio buttons.
`f:selectItem`	Specifies an item in an `h:selectOneMenu` or `h:selectOneRadio` (and other similar components).



Faces Servlet



Faces Servlet

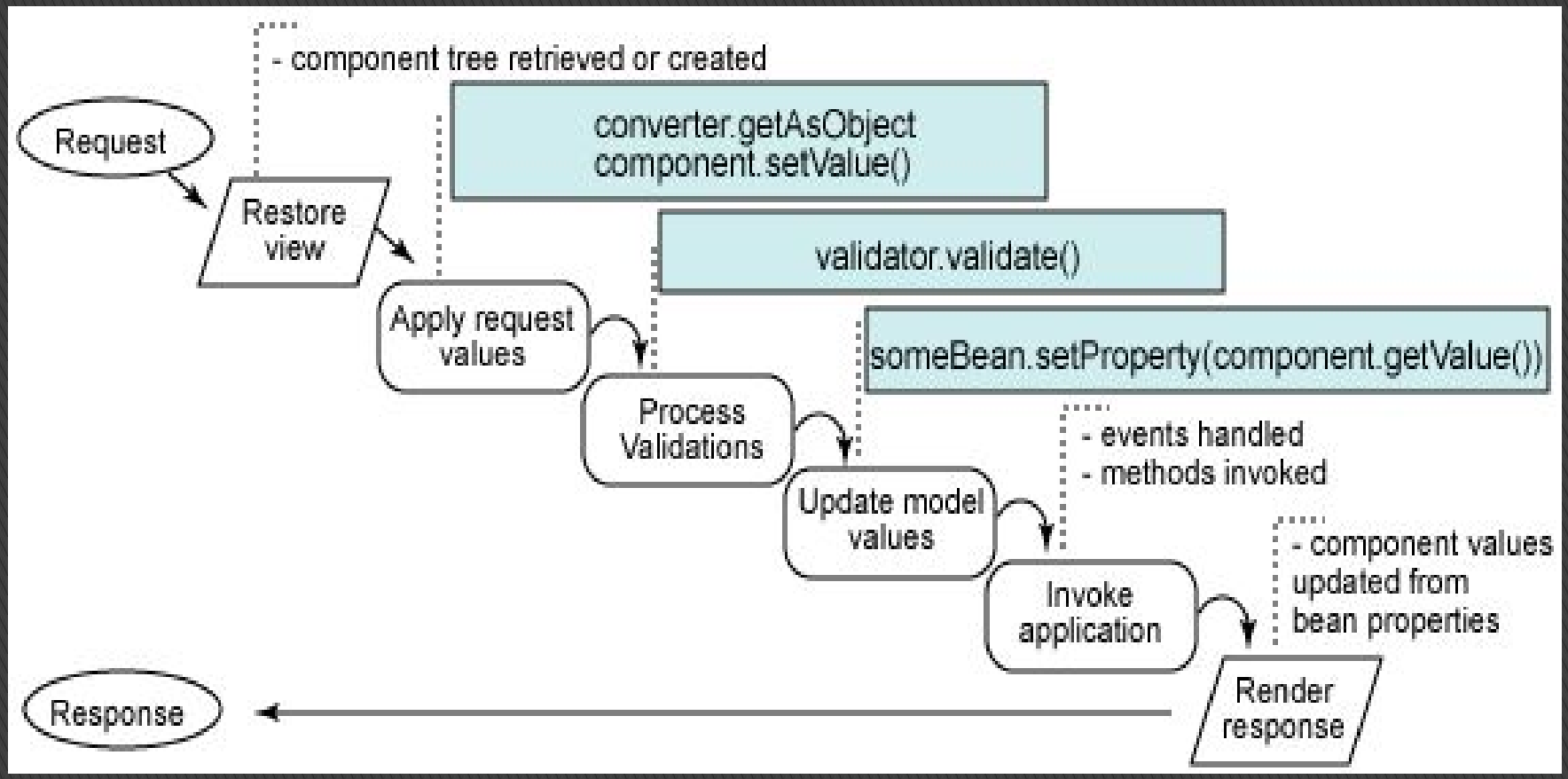
Servlet: A **servlet** egy olyan Java objektum, amely HTTP kérést dolgoz fel és HTTP választ generál – dinamikus tartalomgenerálás

Faces Servlet:

- JSF életciklus vezérlése
- Kérések feldolgozása
- view template betöltés és komponensfa építés
- eseménykezelés
- válasz generálás (többnyire HTML vagy XHTML formátumban
- a felület komponenseit minden lekérés végén elmenti (stateSaving), majd ugyanazon view következő előállításakor újra betölti.
- FacesServlet regisztrálása web.xml–ben:

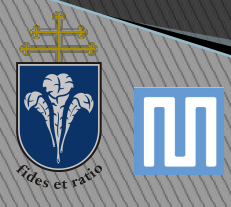
```
<servlet>
    <servlet-name>FacesServlet</servlet-name>
   <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>FacesServlet</servlet-name>
    <url-pattern>/faces/*</url-pattern>
</servlet-mapping>
```

JSF életciklus



Managed Bean

- Managed Bean (backing bean) = Szerver oldali, a felhasználói felülettel összekapcsolt POJO objektum (= nem EJB), az MVCből a Modelt + a Controller logika testre szabását reprezentálják
- faces-config.xml-ben, vagy annotációval adható meg
- @ManagedBean(name="myBean")
- A név alapján érjük el EL-ből (ha nem adjuk meg, az osztály neve, kis kezdőbetűvel)
- Az állapotot reprezentáló property-ket, vagy komponenspéldányt tárolhatunk benne



Managed Bean

- A komponenshez tartozó eseménykezelő, validáló, illetve az oldal-navigációt vezérlő metódusok is elhelyezhetők benne
- A JSF oldalon Expression Language kifejezésként hivatkozhatunk a managed bean property-re vagy metódusra
- pl.:

```
<h:inputText value="#{numberBean.value}"
validator="#{numberBean.validate}" />
```

Managed Bean Scope

- **@NoneScoped –>** minden hivatkozásnál létrejön ha injektáljuk, élettartama megegyezik a tulajdonos bean–ével
- @RequestScoped – egy http request–ig él
- @ViewScoped: request és session közti élettartam,amíg el nem navigálunk az oldalról, AJAX–os kéréseknél hasznos
- @SessionScoped – a session egész élettartama alatt elérhető
- @ApplicationScoped – az alkalmazás futása alatt elérhető

Eseménykezelés

- Events (események): ActionEvent, ValueChangeEvent

ActionEvent automatikusan generálódik h:commandButton megnyomásakor

- EventListeners: pl actionListener () vagy action()
- void actionListener() – actionListener attribútumra kötve
- String action() – Stringben visszaadjuk annak az oldalnak a nevét, ahova át szeretnénk irányítani a felhasználót a függvény lefutása után – action attribútumra kötve

```
<h:form>
    …
     <h:commandButton value="Submit"
actionListener="#{testBean.submit}"/>
  <h:commandButton value="Submit"
action="#{testBean.submitAndNavigate}"/>
```

```
public void submit() {  // handle form submission
    System.out.println ("You submitted: " + input);
}
public String submitAndNavigate() {  // handle form submission
    System.out.println ("You submitted: " + input);
        return „loginPage”; }
```