JAVA SERVER FACES
MANAGED BEANS

SADRZAJ-5

- Java beans
- Java klase koje poštuju sljedeće konvencije:
  - imaju default-ni (zero-argument) konstruktor
  - nemaju public polja
  - imaju setere i getere – setXxx i getXxx metode za pristup i promjenu vrijednosti polja
- često se, u JSF terminologiji, nazivaju pozadinskim bean-ovima – "backend beans"

- Java beans – trebali bi biti serijalizabilni

```
public class TestBean implements Serializable
```

- neki serveri podržavaju distribuirane Web aplikacije
  - pomoću load balancing mehanizma različiti zahtjevi se mogu poslati različitim serverima – sesije bi trebale raditi, bez obzira koji server obrađuje zahtjev
- neki serveri podržavaju perzistentne sesije
  - podaci o sesiji se pamte na disku i ponovo se učitavaju u slučaju da je server restartovan – ovo treba da radi dok god je web čitač korisnika aktivan
  - Tomcat 5+ podržava perzistentne sesije

- Expression Language
- dobre osobine:
  - kraća notacija za pristup property-ima bean-a
    - npr. #{company.companyName} – companyName property, getter getPropertyName() scoped varijable (objekat smješten u request, session ili application scope) ili managed bean-a
    - #{company.president.firstName} – firstName poperty objekta president, koji je property scoped varijable ili managed bean-a Company
  - jednostavan pristup elementima kolekcije
    - za referenciranje elementa niza, List-e ili Map-e koristi se #{variable[indexOrKey]}

dobre osobine:

- jednostavan pristup parametrima zahtjeva, cookie-ima, i drugim podacima zahtjeva
- za pristup standardnim tipovima podataka zahtjeva, može se koristiti jedan od nekoliko predefinisanih implicitnih objekata
- mali, ali koristan skup jednostavnih operatora
  - za obradu objekata pomoću EL izraza, može se koristiti neki od nekoliko aritmetičkih, relacionih, logičkih, ili drugih operatora
- uslovni izlaz
  - za izbor između više izlaznih opcija, ne moraju se koristiti Java skripleti, već #{test ? option1 : option2}.
- automatska konverzija tipova
  - EL uklanja potrebu za konverzijom tipova
- prazna vrijednost umjesto poruke o grešci
  - u većini slučajeva, greška ili NullPointerExceptions daju prazan string, a ne izuzetak

- čitanje property-ja bean-a
- #{varName.propertyName}
  - pretražuje se HttpServletRequest, HttpSession, ServletContext i managed bean-ovi, u ovom poretku, i vraća vrijednost property-ja
  - mora se koristiti u atributu JSF taga
- ekvivalentne forme

```
<h:outputText value="#{customer.firstName}"/>
```

  - radi u svim JSF verzijama - scoped varijable i managed bean-ovi

```
${customer.firstName}
```

  - radi samo u JSP 2.0 i kasnijim verzijama - samo scoped varijable

```
<%@ page import="net.etfbl.NameBean" %>
<% NameBean person =
(NameBean)pageContext.findAttribute("customer"); %>
<%= person.getFirstName() %>
```

  - pre-EL verzija

- čitanje property-ja bean-a

  `#{varName.property1.property2}`
- prvo se izvršava pretraga za odgovarajuću definiciju beana po imenu varName
- nakon toga se pristupa property-iju property1 (poziva se metoda getProperty1())
- nakon toga se pristupa property-iju property2 dobijenog rezultata - poziva se metoda getProperty2() objekta koji je dobijen sa getProperty1()

- tri značenja #
- označavanje izlaznih vrijednosti:

  `#{varName.propertyName}`
  - prikazuje vrijednost property-ija date promenljive

  `<h:outputText value="#{employee.address}"/>`
  - bilo kada da se pristupa, podrazumijeva izlazni tekst

  `<h:inputText value="#{employee.address}"/>`
  - kada se forma inicijalno prikazuje, prikazuje se predefinisana vrijednost
- označavanje submit-ovane vrijednosti

  `<h:inputText value="#{employee.address}"/>`
  - kada se forma šalje, definiše gdje se smješta vrijednost
- dizajn poziva metoda kod slanja

```
<h:commandButton value="Button Label"
  action="#{employee.processEmployee}"/>
```
  - kada se forma šalje, definiše se određena akcija

- pristup kolekcijama
- radi za
  - nizove – ekvivalentno sa

    `theArray[index]`
  - liste – ekvivalentno sa

    `theList.get(index) ili theList.set(index, submitted-val)`
  - mape – ekvivalentno sa

    `theMap.get(key) ili theMap.put(key, submitted-val)`
- ekvivalentne forme (za HashMap-e)
  - #{name.property}
  - #{name["property"]}

- predefinisane varijable
- facesContext – FacesContext objekat
  - #{facesContext.externalContext.session.id}
- param i paramValues – request parametri
  - #{param.custID}
- header i headerValues – request header-i
  - #{header.Accept} or #{header["Accept"]}
  - #{header["Accept-Encoding"]}
- cookie – Cookie objekat
  - #{cookie.userCookie.value} ili #{cookie["userCookie"].value}
- initParam – Context initialization parametar
- requestScope, sessionScope, applicationScope

- potencijalni problem
  - implicitni objekti (predefinisane varijable) obično lošije rade sa MVC modelom

- operatori
- aritmetički
  - + - * / div % mod
- relacioni
  - == eq != ne < lt > gt <= le >= ge
- logički
  - && and || or ! Not
- empty
  - empty
  - vraća true za null, prazan string, prazan niz, praznu listu, praznu mapu
- napomena
  - koristiti ograničeno kako bi MVC model bio očuvan

- ${ test ? expression1 : expression2 }
  - izračunava test i vraća ili expression1 ili expression2
- problemi
  - podrazumijeva stavljanje dijela business logike u JSP stranicu
  - trebalo bi ga koristiti samo za prezentacionu logiku
- napomena
  - koristiti ograničeno kako bi MVC model bio očuvan

JSP Beans Scopes

- Page
- Request
- Session
- Application

JSF managed Beans Scopes

- View
- Flex
- Custom
- Request
- Session
- Appliocation

- .properties datoteka
  - sadrži parove ključ – vrijednost
  - mora biti smještena u WEB-INF/classes
- učitavanje datoteke – pomoću f:loadBundle
  - basename atribut – ime datoteke, bez ekstenzije .properties
  - var – vraća scoped varijablu – Map-u
    - relativno u odnosu na WEB-INF/classes, .properties se podrazumijeva
    - primjer 1, WEB-INF/classes/messages.properties

```
<f:loadBundle basename="messages" var="msgs"/>
```

    - primjer 2., WEB-INF/classes/package1/test.properties

```
<f:loadBundle basename="package1.test" var="msgs"/>
```

- ispis poruka – korišćenjem EL

```
#{msgs.keyName}
```

- parametrizovani stringovi
- kreirati .properties datoteku unutar WEB-INF/classes
  - vrijednosti sadrže {0}, {1}, {2}, itd.
  - npr. poruka=Unesena vrijednost se nalazi u intervalu između {0} i {1} !!!
- učitavanje datoteke – pomoću f:loadBundle
  - basename atribut – ime datoteke, bez ekstenzije .properties
  - var – vraća scoped varijablu – Map-u
- ispis poruka – korišćenjem h:outputFormat
  - vrijednost vraća osnovnu poruku
  - ugnježdeni f:param vraćaju vrijednosti koje se mijenjaju

```
<h:outputFormat value="#{msgs.poruka}">
  <f:param value="vrijednost za ulaz 0"/>
  <f:param value="vrijednost za ulaz 1"/>
</h:outputFormat>
```

- internacionalizacija
- kreirati višestruke .properties datoteke
  - messages_en.properties, messages_sr.properties, messages_hr.properties
- proslijediti locale argument u f:view, putem locale atributa

```
<f:view
  locale="#{facesContext.externalContext.request.locale}">
```

  - pročitati locale iz podešavanja Web čitača

  ili

  - postaviti izabrani locale

```
locale="#{settings.selectedLocale}"
```

- učitavanje datoteke – pomoću f:loadBundle
  - basename atribut – ime datoteke, bez ekstenzije .properties
    - verzija koja odgovara izabranom locale-u će biti automatski korištena
  - var – vraća scoped varijablu – Map-u
- ispis poruka – korišćenjem h:outputForma