Programski Sistemi

Arhitektura Web aplikacija

UNIVERZITET PRVI AKREDITOVANI PRIVATNI UNIVERZITET SINGIDUNUM

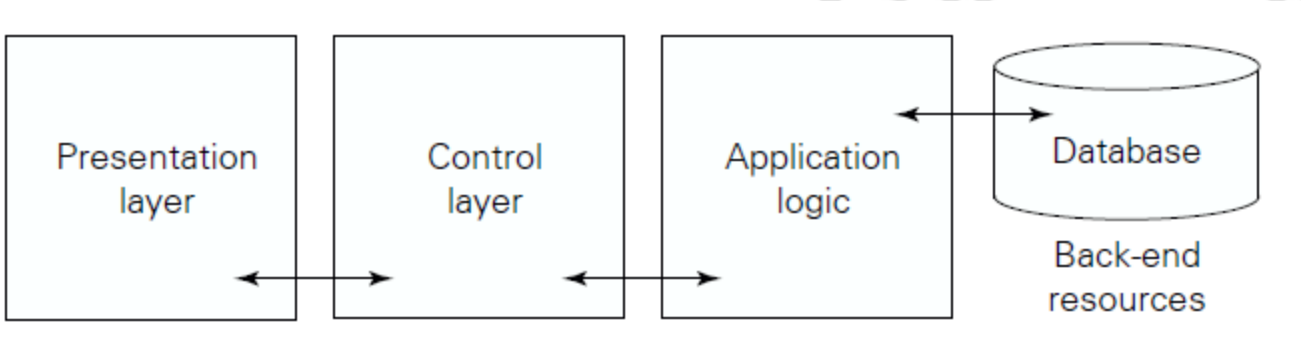
Model View Controller (MVC)

- Kada se projektuje Web aplikacija, njenu arhitekturu je najpogodnije podeliti na tri logičke oblasti:
  - Sloj prezentacije koji kontroliše vizuelni aspekt i prikazuje rezultate, poznat kao **View**
  - Kontrolni sloj koji kontroliše tok aplikacije, poznat kao **Controller**
  - Sloj aplikacione logike koji upravlja podacima, vrši izračunavanja i komunicira sa bazom podataka, poznat kao **Model**

PRVI AKREDITOVANI PRIVATNI UNIVERZITET SINGIDUNUM

MVC

- Svaki sloj ne mora neophodno da bude implementiran kao posebna komponenta
- Neki ili svi slojevi mogu da se kombinuju u jedinstvene komponente da bi se umanjila složenost aplikacije, na račun modularnosti i apstrakcije visokog nivoa



Arhitektura Web aplikacije

- Najvažnija odluka prilikom projektovanja Web aplikacije jeste odluka kako razdvojiti odgovornosti prezentacione, kontrolne i aplikacione logike
- Postoje dva osnovna pristupa:
  - page-centric
  - servlet-centric

UNIVERZITET PRVI AKREDITOVANI PRIVATNI UNIVERZITET SINGIDUNUM

Page-centric dizajn

```
<%
String src = request.getParameter("srcAccount");
String dest = request.getParameter("destAccount");
String amount = request.getParameter("amount");

int result = BankManager.instance().transfer(src, dest, amount);
if (result == 0) {
  response.sendRedirect("transferSuccessful.jsp?amount=" + amount);
}
else {
  String msg;
  if (result == 100)
    msg = "Insufficient Funds";
  else if (result == 200)
    msg = "Destination Account Invalid";
  else if (result == 300)
    msg = "Source Account Invalid";
  else
    msg = "Unknown Error: Transfer Failed";
  // encode the msg for for use as a request parameter
  response.sendRedirect("transferFailed.jsp?msg=" + msg);
}
%>
```

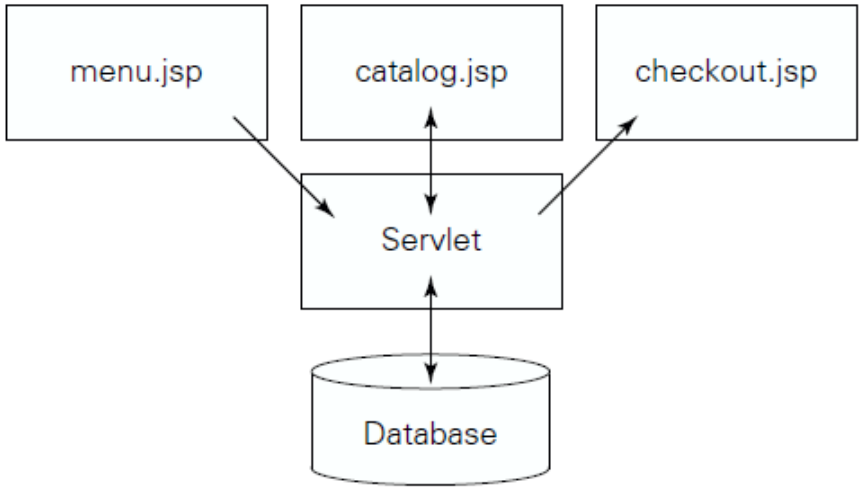
Putanja response.sendRedirect()

- Za razliku od RequestDispatchera i taga <jsp:include>, putanja referencirana u metodi response.sendRedirect() je apsolutna u odnosu na document root servera, a ne kontekst servleta
- Ako želimo da preusmerimo klijenta na neki drugi dokument unutar Web aplikacije, moramo putanju da započnemo kontekstom, na primer:

```
<%response.sendRedirect(request.getContextPath() + "/destination.jsp");%>
```

Servlet-centric dizajn

- Moguće je implementirati MVC pomoću objekta RequestDispatcher i dobiti sasvim zadovoljavajuće rezultate za jednostavne i srednje složene aplikacije
- MVC realizovan pomoću RequestDispatchera je servlet-centric dizajn



MVC pomoću RequestDispatchera (1)

1. Definiše se model za prezentaciju podataka (npr. beanovi)
2. Koristi se servlet za prihvatanje podataka (čitanje parametara zahteva, proveru podataka i sl.)
3. Isti servlet obavlja i poslovnu logiku i postavlja svojstva beanova iz tačke 1 (Controller)
4. Servlet poziva setAttribute u okviru objekata request, session, ili sadržaja servleta da bi smestio beanove koji su rezultat zahteva

MVC pomoću RequestDispatchera (2)

5. Zahtev se prosleđuje JSP strani
   - Servlet prepoznaje koja JSP stranica odgovara datoj situaciji i koristi metod forward objekta RequestDispatcher da bi preusmerio kontrolu datoj stranici
6. JSP strana preuzima podatke iz beana pomoću jsp:useBean i odgovarajuće oblasti važenja iz koraka 4
   - Stranica tada koristi jsp:getProperty da bi pristupila svojstvima beana
   - JSP stranica ne kreira ili modifikuje bean; ona samo prihvata i pokazuje podatke koje servlet kreira (View)

MVC pomoću RequestDispatchera

- Mnoge servlet-centrične aplikacije koriste Command design pattern
- Svaki zahteva sa JSP strane sadrži neku vrstu identifikatora komande, koji pokreće određenu akciju servleta ili na neki drugi način upravlja tokom aplikacije
- Svaka komanda se kapsulira u sopstvenu klasu, čime se funkcionalnosti izdvajaju iz koda kontrolnog servleta

PRVI PRIVATNI UNIVERZITET AKREDITOVANI SINGIDUNUM

MVC pomoću RequestDispatchera (4)

```
String cmd = req.getParameter("cmd");
if (cmd.equals("save")) {
  SaveCommand saver = new SaveCommand();
  saver.save(); // do its thing
}
if (cmd.equals("edit")) {
  EditCommand editor = new EditCommand();
  editor.edit(); // do its thing
}
if (cmd.equals("remove")) {
  RemoveCommand remover = new RemoveCommand();
  remover.remove(); // do its thing
}
```

MVC pomoću RequestDispatchera

- Da bi se izbegao veliki broj if/else blokova, može se primeniti Factory pattern, ili komanda Class.forName()

```
Command cmd = CommandFactory.getCommand(request.getParameter("command"));
cmd.execute();
```

```
String cmdID = request.getParameter("command"));
Command cmd = Class.forName(cmdID + "Command").newInstance();
```

jsp:useBean i MVC nasuprot samo JSP

- JSP stranice ne bi trebalo da kreiraju objekte
  - Servlet, a ne JSP stranica, trebalo bi da kreira sve objekte koji predstavljaju podatke.
  - Da bi se garantovalo da se u okviru JSP stranice neće kreirati objekti trebalo bi koristiti
    <jsp:useBean ... type="package.Class"/>
    umesto
    <jsp:useBean ... class="package.Class" />
- JSP stranica ne bi trebalo da menja objekte: ne treba koristiti jsp:setProperty

RequestDispatcher

- Pomoću metode forward RequestDispatchera:
  - Kontrola se nepovratno predaje novoj stranici
  - Originalna stranica ne može generisati bilo kakav izlaz
- Pomoću metode include RequestDispatchera:
  - Kontrola se trenutno predaje novoj stranici
  - Originalna stranica može generisati izlaz pre i posle uključene stranice
  - Originalni servlet ne vidi izlaz uključene stranice
  - Korisno za portale: JSP prikazuje delove, ali se delovi uređuju različito za različite korisnike

RequestDispatcher: include

```
response.setContentType("text/html");
String firstTable, secondTable, thirdTable;
if (someCondition) {
firstTable = "/WEB-INF/Sports-Scores.jsp";
secondTable = "/WEB-INF/Stock-Prices.jsp";
thirdTable = "/WEB-INF/Weather.jsp";
} else if (...) { ... }
RequestDispatcher dispatcher =
request.getRequestDispatcher("/WEB-INF/Header.jsp");
dispatcher.include(request, response);
dispatcher =
request.getRequestDispatcher(firstTable);
dispatcher.include(request, response);
dispatcher =
request.getRequestDispatcher(secondTable);
dispatcher.include(request, response);
dispatcher =
request.getRequestDispatcher(thirdTable);
dispatcher.include(request, response);
dispatcher =
request.getRequestDispatcher("/WEB-INF/Footer.jsp");
dispatcher.include(request, response);
```

Napredne tehnike

- Korišćenje nekog frameworka, npr.
  - JSF
  - Struts
  - Spring
  - Tapestry
- Vrlo često zahtevaju isto vreme za savladavanje koliko je potrebno da bi se savladali servleti/JSP

PRVI AKREDITOVANI PRIVATNI UNIVERZITET SINGIDUNUM