Web programiranje-teorija

Drage kolege,

Ovde vam se nalazi lista od 20 (možda nešto manje) pitanja (i odgovora), koja je profesor postavljao na ispitima (kako usmenim, tako i pismenim). Što se tiče pismenih ispita, on je u svih 4-5 rokova davao identična pitanja, iste strukture, sa nekim manjima varijacijama, tako da pretpostavljam da će vam tako nešto doći i na kolokvijum. Sve u svemu, od svih onih prezentacija, treba vam sve ukupno 10 slajdova.

1. Kao prvo pitanje uvek je dolazilo ovako nešto: dat je HTML kod na klijentu i verzija HTTP protokola. Treba napisati prvi red HTTP zahteva.
Primer:
<form method="post" action="http://localhost/FormServlet">
<input type="text" name="tekst">
<input type="submit" value="Posalji">
</form>
Napisati prvi red HTTP zahteva u HTTP protokolu verzije 1.1.
Odgovor:
POST / HTTP/1.1
Samo objašnjenje nalazi se u prezentaciji "http_protokol", slajd 6.

2. Neko pitanje vezano za HTTP odgovor.
Primer 1:
Šta se nalazi u HTTP odgovoru neposredno pre sadržaja datoteke?
Odgovor:
Trik pitanje. Prazan red.
Samo objašnjenje nalazi se u prezentaciji "http_protokol", slajd 11.
Primer 2:
Napisati prvi red HTTP odgovora u HTTP protokolu verzije 1.0 za kod 404.
Odgovor:
Obično da poznate kodove (200 i 404), a možda čak i napiše tekstualni opis, ne sećam se.
HTTP/1.0 404 Not Found
Samo objašnjenje nalazi se u prezentaciji "http_protokol", slajd 11.

3. Navesti metode HTTP zahteva i šta radi koji. Ovo pitanje obavezno dolazi. E sad moguće da će pitati samo za jednu metodu ili samo da se navedu, ali u svakom slučaju treba naučiti ceo taj slajd.
Odgovor:

- GET – zahteva resurs od web servera
- POST – šalje parametre forme i traži odgovor
- HEAD – zahteva samo HTTP odgovor (response), bez slanja samog resursa
- PUT – omogućava klijentu da pošalje datoteku na web server

- OPTIONS – od web servera se traži spisak metoda koje podržava
- DELETE – omogućava klijentu da obriše resurs sa web servera

Samo objašnjenje nalazi se u prezentaciji "http_protokol", slajd 7.

4. Navesti atribute i njihovu namenu u HTTP odgovoru. Moguće da pita i za atribute HTTP zahteva, ali čitsto sumnjam. Ako ste baš savesni možete pročitati u prezentaciji "http_protokol", slajd 8. i slajd 9.. Dakle, atributi HTTP odgovora.

Odgovor:

- Content-type – definiše tip odgovora
- Cache-Control – definiše kako se keš na klijentu ažurira
- Location – definiše novu adrese kod redirekcije
- Connection – postoji u HTTP1.1; potvrda klijentu da li da zatvori konekciju ili da je ostavi otvorenu

Samo objašnjenje nalazi se u prezentaciji "http_protokol", slajd 12.

5. Kako se ostvaruje trajna konekcija (*permanent connection*) u HTTP protokolu verzije 1.1?

Odgovor:

Postavljanjem atributa "*Connection*" HTTP odgovora na *"Keep-Alive".*

Ovaj odgovor ne piše nigde, konkretno. Ja sam išla kod njega na pregled radova, pa mi rekao.

6. Koji atributi HTTP zahteva i HTTP odgovora omogućuju praćenje sesije?

Odgovor:

U HTTP zahtevu, atribut koji omogućava praćenje sesije, jeste atribut "*Cookie*", koji čuva primljeni *cookie* i šalje ga uz svaki zahtev. U HTTP odgovoru, atribut koji omogućava praćenje sesije jeste atribut "*Set-Cookie*". Neophodno je navesti i ostale, opcione atribute u HTTP odgovoru, koji su vezani za praćenje sesije, a to su:

- domain – domen u kome važi *cookie*
- path – za koje URL-ove na sajtu važi *cookie*
- expires – datum isticanja *cookie*-a

Samo objašnjenje nalazi se u prezentaciji "pracenje_sesije", slajd 2. i slajd 5..

7. Kako se vrši praćenje sesije, ako navigator ne prihvata *cookie*-e?

Odgovor:

Tada se koristi *URL Rewriting* mehanizam. U hiperlink (<a href="...">) koji "gađa" naš server ugradi se ID sesije: <a href=http://www.myserver.com/catalog/index.html;jsessionid=1234> (ovaj primer sa URL-om obavezno navesti, napisati bilo šta, ali je bitan taj parameter jsessionid=neki brojevi). Moguće je da će pitati i koje metode se koriste pri primeni *URL Rewriting* mehanizma, a to su: HTTPServletResponse.encodeURL() i HTTPServletResponse.encodeRedirectURL() metoda.

Samo objašnjenje nalazi se u prezentaciji "pracenje_sesije", slajd 7..

8. Koje metode ima klasa, koja služi za praćenje sesije i čemu one služe?

Odgovor:

U pitanju je klasa *HttpSession* i ima sledeće metode:

- *getId*() - čuva *cookie* ili ID sesije za *URL Redirection*

- *getAttribute*(ime), *setAttribute*(ime, objekat), *removeAttribute*(ime) - čuva objekte vezane za sesiju

- *invalidate*() - invalidira sesiju i razvezuje sve objekte vezane za nju
- *setMaxInactiveInterval*(sekunde) - podešava period neaktivnosti

Samo objašnjenje nalazi se u prezentaciji "pracenje_sesije", slajd 12..

9. Koju klasu nasledjuju svi servleti i koja metoda te klase se koristi za inicijalizaciju/"uništavanje".
Odgovor:
Klasa *HttpServlet*. Metoda za inicijalizaciju prilikom pokretanja je *HttpServlet.init*(), a metoda za *clean-up* neposredno pre uništavanja je *HttpServlet.destroy*().
Samo objašnjenje nalazi se u prezentaciji "servleti", slajd 7. i slajd 8..

10. Koja metoda klase *HttpServlet* se redefiniše za obradu GET zahteva i koje parametre ona ima?
Odgovor:
Metoda *HttpServlet.doGet*(*HttpServletRequest request, HttpServletResponse response*). Ovde može pitati i za druge zahteve (POST, PUT, HEAD…), a parametri su uvek isti, jedino se razlikuje naziv metode *doGet*, *doPost*, *doPut*...
Samo objašnjenje nalazi se u prezentaciji "servleti", slajd 9..

11. Koje metode sadrži klasa *HttpServletRequest/HttpServletResponse* i čemu one služe?
Odgovor:
Klasa *HttpServletRequest* ima metode: *getParameter*(ime), *getParameterNames*(), *getParameterMap*(), koje služe za izdvajanje parametara GET/POST metode iz forme i smeštanje istih u asocijativnu listu, kao i metode *getHeader*(ime), *getHeaderNames*() i *getHeaders*(ime), koje služe za izdvajanje parametara iz zaglavlja HTTP zahteva i smeštanje istih u asocijativnu listu.
Klasa *HttpServletResponse* ima sledeće metode:
- *setContentType*(vrednost) - čuva tip odgovora
- *addCookie*(cookie) - čuva *cookie*
- *sendRedirect*(nova_lokacija) - omogućuje redirekciju
- *setHeader*(naziv, vrednost) - podešava proizvoljan atribut zaglavlja
- *encodeURL*(url) i *encodeRedirectURL*(url)- ugrađuje ID sesije ako *cookies* nisu uključeni

Samo objašnjenje nalazi se u prezentaciji "servleti", slajd 17. i slajd 21..

12. Navesti vrste dinamičkih elemenata u JSP.
Odgovor:
- izrazi (*expressions*): **<%= *java_izraz* %>**
  <%= new java.util.Date() %>
- skriptleti (*scriptlets*): **<% *java_kod* %>**
  <% for (int i = 0; i < 10; i++) ... %>
- deklaracije (*declarations*): **<%! java_deklaracija %>**
  <%! int a; %>
- direktive (*directives*): **<%@ direktiva attr="..."%>**
  <%@ page contentType="text/plain" %>

Samo objašnjenje nalazi se u prezentaciji “1_JSP”, slajd 7..

13. Navesti i opisati opsege vidljivosti komponenti, te dati primer za jednu od njih.
Odgovor:

- *application* - istu instancu beana dele svi korisnici sajta
- *session* - svaki korisnik sajta ima svoju instancu
- *request* - svaki zahtev za stranicom ima svoju instancu
- *page* (*default*) - svaka stranica ima svoju instancu

<jsp:useBean id="user" class="somepackage.User" scope="**session**"/>
Samo objašnjenje nalazi se u prezentaciji “1_JSP”, slajd 27. i slajd 28..

14. Gde se podešava opseg vidljivosti komponenti?
Odgovor:
U <jsp:useBean> elementu u tagu „*scope*“.
Samo objašnjenje nalazi se u prezentaciji “1_JSP”, slajd 28..

15. Redirekcija u MVC modelu 2.
Odgovor:
Za redirekciju se koristi klasa *RequestDispatcher*, koja ima dve metode: *forward*, koja se koristi kada kompletno prebacimo kontrolu na odredišnu stranicu i *include*, koja se koristi kada ubacujemo sadržaj odredišne stranice i nastavljamo sa tekućom. *RequestDispatcher* omogućuje da na odredištu vidimo request i response objekte iz polazne stranice (servleta).
RequestDispatcher disp = *request.getRequestDispatcher*(relativanURL);
RequestDispatcher disp = *getServletContext().getRequestDispatcher*(apsolutniURL);
Samo objašnjenje nalazi se u prezentaciji “2_MVC”, slajd 16..

16. Napisati deo koda, koji služi za prenos parametara iz servleta u JSP stranicu u zahtevu/sesiji/aplikaciji.
Odgovor:
U servletu:
BeanClass value = new *BeanClass*(...);
request.setAtrribute("bean", value);
U JSP strani:
<jsp:useBean id=”bean” type=”BeanClass” scope=”request”/>
<jsp:getProperty name="bean" property="someProperty" />
Jedino treba voditi računa o *scope*-u, koji može biti request/session/application.
Samo objašnjenje nalazi se u prezentaciji “2_MVC”, slajd 18., slajd 19., slajd 20..

17. Dat je neki JSTL i pitanje je šta je na izlazu. Ponudjeni su odgovori pod a, b i c. Uvek bude neki ovakav kod:

```
<ul> <c:forEach var="message" items="${messages}">
<c:if test="${fn:containsIgnoreCase(message, param.filter) or empty param.filter}">
<li><b><c:out value="${message}"/></b></li>
</c:if>
</c:forEach>
</ul>
```

```
<form action="test2.jsp">
 <input type="text" name="filter">
 <input type="submit" value="Filtriraj">
</form>
```

Ponudjeni odgovori budu obicno: a) ništa, b) neka_reč neka_reč neka_reč c) neka_reč neka_reč neka_reč neka_reč… U kodu, naravno, uvek bude neki for/foreach.

Bilo je još neko pitanje tipa “Gde se podešava *boundary* element”, ali nikad nisam znala odgovor na njega, niti sam našla u prezentacijama.

Toliko od mene.

Pozdravljam vas ☺