```
 1 import java.io.*;
 2 import java.sql.*;
 3 import javax.servlet.*;
 4 import javax.servlet.http.*;
 5
 6 public class TeeVaraus extends HttpServlet {
 7
 8   final String dbDriver="org.postgresql.Driver";
 9   final String dbServer ="jdbc:postgresql://localhost:10388/tsoha";
10   final String dbUser= "avtanska";        // replace with your db user account
11   final String dbPassword ="postgres"; // replace with your password
12
13
14   public void service(HttpServletRequest req, HttpServletResponse res)
15     throws ServletException, IOException {
16     ServletOutputStream out;
17     res.setContentType("text/html");
18     out= res.getOutputStream();
19
20     String submit = req.getParameter("submit");
21     HttpSession session = req.getSession(false);
22     String sessionTunnus = (String)session.getValue("sessionTunnus");
23
24     String paiva = req.getParameter("paiva");
25     String kuukausi = req.getParameter("kuukausi");
26     String vuosi = req.getParameter("vuosi");
27     String aihe = req.getParameter("aihe");
28     String henkilo = req.getParameter("henkilo");
29     String ryhma = req.getParameter("ryhma");
30     String alkuaika = req.getParameter("alkuaika");
31     String kesto = req.getParameter("kesto");
32     String nakyvyys = req.getParameter("nakyvyys");
33     String virhe = "";
34
35     /*
36      * Asetetaan tyhjät arvot
37      */
38
39     if (submit != null) {
40       if (henkilo.equals("tyhjä")) {
41         henkilo = null;
42       }
43       if (ryhma.equals("tyhjä")) {
44         ryhma = null;
45       }
46       if (nakyvyys.equals("tyhjä")) {
47         nakyvyys = null;
48       }
49       if (paiva.equals("tyhjä") || kuukausi.equals("tyhjä")
50           || vuosi.equals("tyhjä")) {
51         paiva = null;
52       }
53       if (alkuaika.equals("tyhjä") || kesto.equals("tyhjä")) {
54         alkuaika = null;
55       }
56       if (aihe.length() == 0) {
57         aihe = null;
58       }
59
60     }
61
62
63     /*
64      * Luodaan virheilmoitukset puuttuville kentille
65      */
66
67     if (henkilo == null && ryhma == null)
68       virhe += "<span class='virhe'>Valitse henkilö tai ryhmä</span><br />";
69     if (aihe == null)
```

```
70        virhe += "<span class='virhe'>Varauksen aihe puuttuu</span><br />";
71      if (nakyvyys == null)
72        virhe += "<span class='virhe'>Valitse näkyvyys</span><br />";
73      if (paiva == null)
74        virhe += "<span class='virhe'>Valitse päivämäärä</span><br />";
75      if (alkuaika == null)
76        virhe += "<span class='virhe'>Valitse alkuaika ja kesto</span><br />";
77
78
79      /*
80       * Jos tullaan sivulle 'Tee varaus'-linkin kautta, tai lähetetään
81       * lomake puuttuvilla tiedoilla.
82       */
83
84      if (submit == null || (submit != null && virhe.length() > 0)) {
85
86        out.println("<html><head><title>Tee varaus</title>" +
87                    "<script type='text/javascript' src='" +
88                    "http://db.cs.helsinki.fi/u/avtanska/tsoha/tarkista.js'>" +
89                    "</script>" +
90                    "<link rel='stylesheet' style='text/css' " +
91                    "href='http://db.cs.helsinki.fi/u/avtanska/tsoha/" +
92                    "perus.css'></head>" +
93                    "<body bgcolor=white>");
94
95        /*
96         * Tulostetaan virheilmoitus tarvittaessa
97         */
98
99        if (submit != null && virhe.length() > 0) {
100         out.println("<p>" + virhe + "</p>");
101       }
102
103       out.println("<h3>Tee varaus, " + sessionTunnus + "</h3>");
104
105
106       /*
107        * Tulostetaan lomake varauksen tekoa varten
108        */
109
110       out.println("<form method='post' name='varaus' " +
111                   "onSubmit='return tarkista()' " +
112                   "action='http://db.cs.helsinki.fi/s/avtanska/TeeVaraus'>" +
113                   "<table cellpadding='10'><tr><td>" +
114                   "Varauksen aihe:<br /><input name='aihe' type='text'></td>");
115
116
117       /*
118        * Haetaan ryhmät tietokannasta
119        */
120
121       out.println("<td>Ryhmä:<br /><select name='ryhma'>" +
122                   "<option value='tyhjä'>-- valitse --");
123
124       Connection con=null;
125       con= createDbConnection(dbDriver,dbServer,dbUser,dbPassword,out);
126       if (con==null) {
127          out.println("</body></html>");
128          return;
129       }
130
131       Statement stmt = null;
132       ResultSet rs = null;
133       try {
134           stmt = con.createStatement();
135           rs = stmt.executeQuery("SELECT nimi FROM ryhmä ORDER BY nimi");
136           while(rs.next()) {
137                out.println("<option>" + rs.getString("nimi"));
138           }
```

```
139
140         } catch (SQLException ee) {
141               out.println("Tietokantavirhe "+ee.getMessage());
142         } finally {
143               try {
144                  if (rs!=null) rs.close();
145                  if (stmt!=null) stmt.close();
146
147               } catch(SQLException e) {
148                  out.println("An SQL Exception was thrown.");
149               }
150         }
151
152         out.println("</select>  tai</td></tr>");
153
154
155         /*
156          * Varauksen päivämäärä
157          */
158
159         out.println("<tr><td>Varauksen päivämäärä:<br />" +
160                     "<select name='paiva'><option value='tyhjä'>DD");
161         for (int i = 0; i < 31; i++) {
162           out.println("<option>" + CheckDate.c(i+1));
163         }
164
165
166         out.println("</select>" +
167                     "<select name='kuukausi'><option value='tyhjä'>MM");
168
169         for (int i = 0; i < 12; i++) {
170           out.println("<option>" + CheckDate.c(i+1));
171         }
172
173         out.println("</select>" +
174                     "<select name='vuosi'><option value='tyhjä'>YYYY" +
175                     "<option>2002<option>2003<option>2004</select>" +
176                     "</td>");
177
178
179         /*
180          * Haetaan henkilöt tietokannasta, admin-käyttäjää ei listaan
181          * tulosteta.
182          */
183
184         out.println("<td>Henkilö:<br /><select name='henkilo'>" +
185                     "<option value='tyhjä'>-- valitse --");
186
187
188         if (con==null) {
189            out.println("</body></html>");
190            return;
191         }
192
193         stmt = null;
194         rs = null;
195         try {
196             stmt = con.createStatement();
197             rs = stmt.executeQuery("SELECT tunnus FROM henkilö ORDER BY tunnus");
198             while(rs.next()) {
199               if (!(rs.getString("tunnus")).equals("admin")) {
200                  out.println("<option>" + rs.getString("tunnus"));
201               }
202             }
203
204         } catch (SQLException ee) {
205               out.println("Tietokantavirhe "+ee.getMessage());
206         } finally {
207               try {
```

```
208                if (rs!=null) rs.close();
209                if (stmt!=null) stmt.close();
210
211             } catch(SQLException e) {
212                out.println("An SQL Exception was thrown.");
213             }
214        }
215
216        out.println("</select></td>");
217
218
219        /*
220         * Varauksen alkuaika
221         */
222
223        out.println("</tr><tr><td>Varauksen alkuaika:<br />" +
224                    "<select name='alkuaika'><option value='tyhjä'>HH:MM");
225
226        for (int i = 0; i < 24; i++) {
227          if (i < 10) {
228            out.println("<option>0" + i + ":00");
229            out.println("<option>0" + i + ":30");
230          }
231          else {
232            out.println("<option>" + i + ":00");
233            out.println("<option>" + i + ":30");
234          }
235        }
236
237        out.println("</select></td>");
238
239
240        /*
241         * Haetaan näkyvyydet tietokannasta
242         */
243
244        out.println("<td>Näkyvyys:<br /><select name='nakyvyys'>" +
245                    "<option value='tyhjä'>-- valitse --");
246
247        stmt = null;
248        rs = null;
249        try {
250            stmt = con.createStatement();
251            rs = stmt.executeQuery("SELECT * FROM näkyvyys");
252            while(rs.next()) {
253                 out.println("<option value='" + rs.getString("nid") +
254                             "'>" + rs.getString("kuvaus"));
255            }
256        } catch (SQLException ee) {
257              out.println("Tietokantavirhe "+ee.getMessage());
258        } finally {
259              try {
260                 if (rs!=null) rs.close();
261                 if (stmt!=null) stmt.close();
262                 con.close();
263              } catch(SQLException e) {
264                 out.println("An SQL Exception was thrown.");
265              }
266        }
267
268        out.println("</select></td></tr>");
269
270
271        out.println("<tr><td colspan='2'>Varauksen kesto:<br/>" +
272                    "<select name='kesto'><option value='tyhjä'>--");
273
274        for (int i = 0; i < 24; i++) {
275          if (i == 0) {
276            out.println("<option>0.5");
```

```
277         }
278         else {
279           out.println("<option>" + i + ".0");
280           out.println("<option>" + i + ".5");
281         }
282       }
283       out.println("<option>24.0");
284
285       out.println("</select> tuntia</td></tr>");
286
287       out.println("<td colspan='2' align='left'>" +
288                   "<input name='submit' type='submit' value='Varaa'>" +
289                   "<input type='reset' name='reset' value='Tyhjennä'></td>" +
290                   "</table></form></body></html>");
291
292
293
294     } // if submit == null
295
296
297
298     /*
299      * Kun lomake lähetetty oikein täytetyillä kentillä
300      */
301
302     else {
303
304
305       out.println("" +
306         "<html><head><title>l</title><link rel='stylesheet' type='text/css'" +
307         "href='http://db.cs.helsinki.fi/u/avtanska/tsoha/perus.css' />" +
308         "</head><body>");
309
310       String pvm = paiva + "." + kuukausi + "." + vuosi;
311
312       /*
313        * Huolehditaan, että ei tehdä varausta esim. 30.2.2002
314        */
315
316       pvm = CheckDate.checkDate(pvm);
317
318
319
320
321
322       Connection con = null;
323       con= createDbConnection(dbDriver,dbServer,dbUser,dbPassword,out);
324
325       if (con==null) {
326          out.println("</body></html>");
327          return;
328       }
329
330       Statement stmt = null;
331       ResultSet rs = null;
332
333
334       /*
335        * Tarkistetaan, mahtuuko yritetty varaus henkilön tai kaikkien
336        * ryhmän henkilöiden kalentereihin
337        */
338
339       boolean mahtuuKalenteriin = true;
340       String virheIlmoitus = "";
341       String varausKatkaistu = "";
342       String selectVaraukset = "";
343       double alkuaikaSaatu = 0;
344       double kestoSaatu = 0;
345
```

```
346
347         /*
348          * Muutetaan yritetyn varauksen ajat laskettavaan muotoon
349          */
350
351         double alkuaikaYritys =
352             (new Double(alkuaika.substring(0,2))).doubleValue();
353
354         if ((new Double("0." + alkuaika.substring(3,5))).doubleValue() == 0.3) {
355           alkuaikaYritys = alkuaikaYritys + 0.5;
356         }
357
358         double kestoYritys = new Double(kesto).doubleValue();
359
360         double uusiKesto = 0;
361
362
363         /*
364          * Jos yritetty varaus menee vuorokausirajan yli, katkaistaan
365          * puolenyön yli menevä osuus pois ja kerrotaan käyttäjälle
366          * asiasta.
367          */
368
369         if (alkuaikaYritys + kestoYritys > 24) {
370           kestoYritys = kestoYritys - ((alkuaikaYritys + kestoYritys) - 24);
371           kesto = "" + kestoYritys;
372           varausKatkaistu =
373             "<span class='virhe'>Varauksesi jatkuu seuraavan" +
374             " päivän puolelle ja tietokantaan<br /> on tallennettu " +
375             " vain varauspäivän puolelle mahtuva osa. <br />Jos haluat jatkaa" +
376             " varausta seuraavaksi päiväksi, ole hyvä<br /> ja tee uusi" +
377             " varaus aamuksi.</span>";
378         }
379
380
381         /*
382          * SQL-lause varauksen tallettamiseen
383          */
384
385         String insertVaraus = "";
386
387         if (henkilo != null) {
388           insertVaraus =
389              "INSERT INTO varaus " +
390                 "(tunnus, pvm, alkuaika, kesto, aihe, näkyvyys, varaaja) " +
391                 "VALUES ('" + henkilo + "', to_date('" + pvm +
392                 "', 'DD.MM.YYYY'), '" + alkuaika + "'," + kesto + ",'" + aihe +
393                 "'," + nakyvyys + ",'" + sessionTunnus + "')";
394         }
395         else if (ryhma != null) {
396           insertVaraus =
397              "INSERT INTO varaus " +
398                 "(pvm, alkuaika, kesto, aihe, näkyvyys, varaaja, ryhmä) " +
399               "VALUES (to_date('" + pvm + "', 'DD.MM.YYYY'), '" + alkuaika +
400               "'," + kesto + ",'" + aihe + "'," + nakyvyys +
401               ",'" + sessionTunnus + "', '" + ryhma  + "')";
402         }
403
404
405
406         if (henkilo != null) {
407
408           /*
409            * Haetaan henkilön kaikki varaukset sille päivälle, jolle
410            * uutta varausta yritetään tehdä.
411            */
412
413           try {
414             stmt = con.createStatement();
```

```
415           selectVaraukset =
416              "SELECT * FROM varaus WHERE tunnus='" +
417              henkilo + "' AND pvm=to_date('" + pvm +
418              "', 'DD.MM.YYYY') ORDER BY alkuaika";
419
420           rs = stmt.executeQuery(selectVaraukset);
421
422           while(rs.next()) {
423
424             /*
425              * Muutetaan käsiteltävän varauksen ajat laskettavaan muotoon
426              */
427
428             alkuaikaSaatu =
429             (new Double((rs.getString("alkuaika")).substring(0,2))).doubleValue();
430
431             if((new Double("0." +
432                (rs.getString("alkuaika")).substring(3,5))).doubleValue() == 0.3) {
433               alkuaikaSaatu = alkuaikaSaatu + 0.5;
434             }
435
436             kestoSaatu = new Double(rs.getString("kesto")).doubleValue();
437
438
439             /*
440              * Tarkistetaan, menevätkö yritetty ja kalenterissa jo oleva varaus
441              * päällekkäin. Päällekkäisyyksiä on kolmea lajia. Kaikille on
442              * omat virheilmoituksensa.
443              */
444
445             if (alkuaikaYritys == alkuaikaSaatu) {
446               mahtuuKalenteriin = false;
447               virheIlmoitus = "Yrittämäsi varaus alkaa samanaikaisesti " +
448                               "vanhan varauksen kanssa.";
449               break;
450             }
451             else if (alkuaikaYritys < alkuaikaSaatu &&
452                      (alkuaikaYritys+kestoYritys) > alkuaikaSaatu) {
453               mahtuuKalenteriin = false;
454               double erotus = (alkuaikaYritys+kestoYritys)-alkuaikaSaatu;
455               virheIlmoitus = "Yrittämäsi varaus on " + erotus +
456                               " tuntia liian pitkä.<br />" +
457                               "Se menee " +
458                               "seuraavan varauksen päälle.";
459               break;
460             }
461             else if (alkuaikaYritys > alkuaikaSaatu &&
462                      (alkuaikaSaatu+kestoSaatu) > alkuaikaYritys) {
463               mahtuuKalenteriin = false;
464
465               double erotus = (alkuaikaSaatu+kestoSaatu)-alkuaikaYritys;
466               virheIlmoitus = "Yrittämäsi varaus alkaa " + erotus +
467                               " tuntia liian aikaisin.<br /> " +
468                               "Edellinen varaus ei ole vielä ehtinyt loppua.";
469               break;
470             }
471             else {
472               mahtuuKalenteriin = true;
473             }
474
475
476           }
477
478         } catch (SQLException ee) {
479              out.println("Tietokantavirhe "+ee.getMessage());
480         } finally {
481              try {
482                 if (rs!=null) rs.close();
483                 if (stmt!=null) stmt.close();
```

```
484
485               } catch(SQLException e) {
486                  out.println("An SQL Exception was thrown.");
487               }
488         }
489
490       } // loppu if henkilon varaus
491
492       else if (ryhma != null) {
493
494         /*
495          * Jos varausta tehdään ryhmälle, haetaan tietokannasta
496          * ryhmän kaikkien jäsenten varaukset sille päivälle,
497          * jolle varausta yritetään tehdä
498          */
499
500         try {
501           stmt = con.createStatement();
502           selectVaraukset =
503              "SELECT * FROM varaus WHERE tunnus IN " +
504              "(SELECT tunnus FROM jäsenyys WHERE ryhmä='" + ryhma +
505              "') AND pvm=to_date('" + pvm +
506              "', 'DD.MM.YYYY') ORDER BY alkuaika";
507
508           rs = stmt.executeQuery(selectVaraukset);
509
510           while(rs.next()) {
511
512             /*
513              * Muutetaan käsiteltävän varauksen ajat laskettavaan muotoon
514              */
515
516             alkuaikaSaatu =
517             (new Double((rs.getString("alkuaika")).substring(0,2))).doubleValue();
518
519             if((new Double("0." +
520                (rs.getString("alkuaika")).substring(3,5))).doubleValue() == 0.3) {
521               alkuaikaSaatu = alkuaikaSaatu + 0.5;
522             }
523
524             kestoSaatu = new Double(rs.getString("kesto")).doubleValue();
525
526
527             /*
528              * Tarkistetaan, menevätkö yritetty ja kalenterissa jo oleva varaus
529              * päällekkäin. Päällekkäisyyksiä on kolmea lajia. Kaikille on
530              * omat virheilmoituksensa.
531              */
532
533             if (alkuaikaYritys == alkuaikaSaatu) {
534               mahtuuKalenteriin = false;
535               virheIlmoitus = "Yrittämäsi varaus alkaa samanaikaisesti " +
536                               "mm. henkilön '" + rs.getString("tunnus") +
537                               "' vanhan varauksen kanssa.";
538               break;
539             }
540             else if (alkuaikaYritys < alkuaikaSaatu &&
541                      (alkuaikaYritys+kestoYritys) > alkuaikaSaatu) {
542               mahtuuKalenteriin = false;
543               double erotus = (alkuaikaYritys+kestoYritys)-alkuaikaSaatu;
544               virheIlmoitus = "Yrittämäsi varaus on " + erotus +
545                               " tuntia liian pitkä.<br />" +
546                               "Se menee mm. henkilön '" +
547                               rs.getString("tunnus") +
548                               "' seuraavan varauksen päälle.";
549               break;
550             }
551             else if (alkuaikaYritys > alkuaikaSaatu &&
552                      (alkuaikaSaatu+kestoSaatu) > alkuaikaYritys) {
```

```
553               mahtuuKalenteriin = false;
554
555               double erotus = (alkuaikaSaatu+kestoSaatu)-alkuaikaYritys;
556               virheIlmoitus = "Yrittämäsi varaus alkaa " + erotus +
557                               " tuntia liian aikaisin.<br /> " +
558                               "Mm. henkilön '" + rs.getString("tunnus") +
559                               "' edellinen varaus ei ole vielä ehtinyt loppua.";
560               break;
561             }
562             else {
563               mahtuuKalenteriin = true;
564             }
565
566
567           }
568
569         } catch (SQLException ee) {
570               out.println("Tietokantavirhe "+ee.getMessage());
571         } finally {
572               try {
573                  if (rs!=null) rs.close();
574                  if (stmt!=null) stmt.close();
575
576               } catch(SQLException e) {
577                  out.println("An SQL Exception was thrown.");
578               }
579         }
580
581
582
583       }
584
585
586
587       /*
588        * Jos varaus mahtuu kalenteriin, viedään sen tiedot tietokantaan.
589        * Jos varaus kuuluu ryhmälle, tehdään varaus myös kaikkien ryhmän
590        * jäsenten kalentereihin.
591        */
592
593       if (mahtuuKalenteriin) {
594
595         stmt = null;
596         rs = null;
597         try {
598             stmt = con.createStatement();
599             int count = stmt.executeUpdate(insertVaraus);
600
601             if (ryhma != null) {
602               rs = stmt.executeQuery(
603                  "SELECT tunnus FROM jäsenyys WHERE ryhmä='" + ryhma + "'");
604               while (rs.next()) {
605                 count = stmt.executeUpdate(
606                    "INSERT INTO varaus " +
607                    "(tunnus, pvm, alkuaika, kesto, aihe, näkyvyys, varaaja) " +
608                    "VALUES ('" + rs.getString("tunnus") + "', to_date('" +
609                    pvm + "', 'DD.MM.YYYY'), '" + alkuaika + "'," + kesto +
610                    ",'" + aihe + "', " + nakyvyys + ",'" + sessionTunnus +
611                    "')");
612               }
613             }
614
615         } catch (SQLException ee) {
616               out.println("Tietokantavirhe "+ee.getMessage());
617         } finally {
618               try {
619                  if (rs!=null) rs.close();
620                  if (stmt!=null) stmt.close();
621                  con.close();
```

```
622               } catch(SQLException e) {
623                  out.println("An SQL Exception was thrown.");
624               }
625          }
626
627
628           String henkilöTaiRyhmä = "";
629
630           if (henkilo != null) {
631             henkilöTaiRyhmä = henkilo;
632           }
633           else if (ryhma != null) {
634             henkilöTaiRyhmä = ryhma;
635           }
636
637           out.println("<h3>Kiitos varauksesta.</h3>");
638
639
640           /*
641            * Huomautetaan käyttäjää, jos varaus on jouduttu katkaisemaan
642            */
643
644           if (!varausKatkaistu.equals("")) {
645             out.println("<span class='virhe'><h3>HUOM!</h3></span>" +
646                         "<p>" + varausKatkaistu + "</p>");
647           }
648
649           out.println("<p>Kenelle: <b>" + henkilöTaiRyhmä + "</b></p>" +
650             "<p>Aihe: <b>" + aihe + "</b></p><p>Pvm: <b>" + pvm +
651             "</b></p><p>Alkuaika: <b>" + alkuaika + "</b></p>" +
652             "<p>Kesto: <b>" + kesto + " tuntia</b></p>");
653
654        } // loppu if mahtuuKalenteriin
655
656
657        /*
658         * Jos varaus ei mahdu kalenteriin
659         */
660
661        else {
662          out.println("<h3>Varaus ei mahdu kalenteriin.</h3>" +
663                      "<p class='virhe'>" + virheIlmoitus + "</p>");
664
665          if (henkilo != null ) {
666            out.println("<p>Tarkista vielä kyseisen henkilön kalenterista " +
667                        "vapaat ajat.</p>");
668          }
669          else if (ryhma != null) {
670            out.println("<p>Tarkista vielä kyseisen ryhmän kaikille " +
671                        "jäsenille yhteiset vapaat ajat<br />ryhmän" +
672                        "hallintasivulla olevan 'Ryhmän ajat'-napin kautta.</p>");
673          }
674
675          double loppuKorkeus =
676            (alkuaikaYritys+kestoYritys < alkuaikaSaatu+kestoSaatu
677             ? alkuaikaSaatu+kestoSaatu : alkuaikaYritys+kestoYritys);
678
679
680          /*
681           * Korjataan liian suuri loppuKorkeus
682           */
683
684          if (loppuKorkeus >= 19) {
685            loppuKorkeus = 19;
686          }
687
688
689          /*
690           * Tulostetaan varauksen tekijälle näkymä, miten yritetty varaus
```

```
691          * menee kalenterissa jo olevan varauksen päälle.
692          */
693
694         out.println("<table border='2' height='" + (loppuKorkeus*20+100) +
695                     "' class='kalenteri'><tr>");
696
697         out.println("<td class='kalenteri' valign='top'>" +
698                     "<table border='0' class='ajat' cellpadding='0' " +
699                     " cellspacing='0'>");
700
701         for (int i = 0; i < (int)loppuKorkeus+5; i++) {
702           out.println("<tr><td class='kalenteri' valign='top' " +
703                       "height='20' style='font-size: 8pt'>"
704                       + CheckDate.c(i) + ":00</td></tr>");
705         }
706
707         out.println("</table></td>");
708
709         out.println("<td class='kalenteri' " +
710                     "valign='top'><table class='kalenteri'><tr><td height='" +
711                     alkuaikaSaatu*20 + "' class='kalenteri' width='100'>" +
712                     " </td></tr><tr>" +
713                     "<td class='varaus' valign='top' height='" +
714                     kestoSaatu*20 + "'><b>Vanha varaus</b></td></tr>" +
715                     "</table></td><td class='kalenteri' valign='top'>" +
716                     "<table class='kalenteri'><tr>" +
717                     "<td class='kalenteri' height='" + alkuaikaYritys*20 +
718                     "' width='100'> </td></tr><tr><td class='yritys' " +
719                     " valign='top' height='" +
720                     kestoYritys*20 + "'><b>Yritetty varaus</b></td>" +
721                     "</tr></table></td></tr></table>");
722       }
723
724       out.println("</body></html>");
725
726     }
727
728   }
729
730
731   private Connection createDbConnection(
732     String dbDriver, String dbServer, String dbUser, String dbPassword,
733     ServletOutputStream out) throws IOException {
734
735     // establish a database connection
736     try{
737         Class.forName(dbDriver);               // load driver
738     } catch (ClassNotFoundException e) {
739           out.println("Couldn't find driver "+dbDriver);
740           return null;
741     }
742     Connection con=null;
743     try {
744        con = DriverManager.getConnection(dbServer,dbUser,dbPassword);
745     } catch (SQLException se) {
746           out.println("Couldn\'t get connection to "+dbServer+
747                       " for "+ dbUser+"<br>");
748           out.println(se.getMessage());
749     }
750     return con;
751   }
752
753
754 }
```