```
 1 import java.io.*;
 2 import java.sql.*;
 3 import javax.servlet.*;
 4 import javax.servlet.http.*;
 5 import java.util.Calendar;
 6 import java.text.DateFormat;
 7 import java.util.Date;
 8
 9
10 public class KalenteriYleinen extends HttpServlet {
11
12   final String dbDriver="org.postgresql.Driver";
13   final String dbServer ="jdbc:postgresql://localhost:10388/tsoha";
14   final String dbUser= "avtanska";        // replace with your db user account
15   final String dbPassword ="postgres"; // replace with your password
16
17
18   public void service(HttpServletRequest req, HttpServletResponse res)
19          throws ServletException, IOException {
20
21     ServletOutputStream out;
22     res.setContentType("text/html");
23     out= res.getOutputStream();
24
25     Calendar kalenteri = Calendar.getInstance();
26
27     HttpSession session = req.getSession(false);
28     String sessionTunnus = (String)session.getValue("sessionTunnus");
29
30
31     /*
32      * Tämä muuttuja määrittelee kuinka monta pikseliä yhden
33      * tunnin korkeus kalenterissa on. HUOM! Jos muutat, muista
34      * vaihtaa myös tuntiviivoja esittävä kuva kale_bg.gif.
35      */
36
37     final int TUNNIN_KORKEUS = 26;
38
39     String henkilo = req.getParameter("henkilo");
40     String ryhma = req.getParameter("ryhma");
41     String alkupvm = req.getParameter("alkupvm");
42     String henkilöTaiRyhmä = "";
43
44
45     if (henkilo != null && henkilo.equals("tyhjä")) {
46       henkilo = null;
47     }
48     if (ryhma != null && ryhma.equals("tyhjä")) {
49       ryhma = null;
50     }
51
52     out.println("<html><head><title>Database query from DB (tsoha)</title>" +
53     "<link rel='stylesheet' style='text/css' " +
54     "href='http://db.cs.helsinki.fi/u/avtanska/tsoha/perus.css'></head>" +
55     "<body>");
56
57
58
59
60     if (ryhma == null && henkilo == null) {
61       out.println("<h3>Valitse ryhmä tai henkilö.</h3>");
62     }
63     else {
64
65
66       if (henkilo != null) {
67         henkilöTaiRyhmä = "henkilo=" + henkilo;
68         out.println("<h3>Varaukset henkilölle '" + henkilo + "'</h3>");
69       }
```

```
70        else if (ryhma != null) {
71          henkilöTaiRyhmä = "ryhma=" + ryhma;
72          out.println("<h3>Varaukset ryhmälle '" + ryhma + "'</h3>");
73        }
74
75        out.println("<form method='post' " +
76           "action='http://db.cs.helsinki.fi/s/avtanska/Kalenteri'>");
77
78
79
80        /*
81         * Alustetaan kalenteri osoittamaan käynnissä olevan
82         * viikon ensimmäiseen päivään (Euroopassa maanantai)
83         */
84
85        DateFormat df = DateFormat.getDateInstance();
86        Date tempPvm = null;
87        int alkuun = 0;
88
89        if (alkupvm == null) {
90          alkuun =
91             kalenteri.getFirstDayOfWeek() - kalenteri.get(Calendar.DAY_OF_WEEK);
92          if (kalenteri.get(Calendar.DAY_OF_WEEK) == Calendar.SUNDAY) {
93            alkuun = -6;
94          }
95          kalenteri.add(Calendar.DATE, alkuun);
96          alkupvm = "" + df.format(kalenteri.getTime());
97        }
98
99        try {
100         tempPvm = df.parse(alkupvm);
101       } catch (Exception e) {
102         out.println("Virhe: " + e);
103       }
104
105       kalenteri.setTime(tempPvm);
106
107
108       /*
109        * Tässä pyöritellään kalenteria edestakaisin, jotta saadaan
110        * oikeat päivämäärät sen tulostusta ohjaaville nuolille, sekä
111        * päivämäärät tulostettavan aikavälin päätepisteille
112        */
113
114       kalenteri.add(Calendar.DATE, -1);
115       String taakse = df.format(kalenteri.getTime());
116
117       kalenteri.add(Calendar.DATE, 2);
118       String eteen = df.format(kalenteri.getTime());
119
120       kalenteri.add(Calendar.DATE, -8);
121       String viikkoTaakse = df.format(kalenteri.getTime());
122
123       kalenteri.add(Calendar.DATE, 14);
124       String viikkoEteen = df.format(kalenteri.getTime());
125
126       kalenteri.add(Calendar.DATE, -7);
127       String viikonAlku = CheckDate.c(kalenteri.get(Calendar.DATE)) +
128                           "." + CheckDate.c(kalenteri.get(Calendar.MONTH)+1) +
129                           "." + kalenteri.get(Calendar.YEAR);
130
131       out.println("<p>Varaukset ajalle: <b>" + viikonAlku + " - ");
132
133       kalenteri.add(Calendar.DATE, 6);
134       String viikonLoppu = CheckDate.c(kalenteri.get(Calendar.DATE)) +
135                            "." + CheckDate.c(kalenteri.get(Calendar.MONTH)+1) +
136                            "." + kalenteri.get(Calendar.YEAR);
137
138       out.println(viikonLoppu + "</b></p>");
```

```
139
140         kalenteri.add(Calendar.DATE, -6);
141
142
143         out.println("<table cellpadding='0' cellspacing='1' " +
144                     "style='background: #000000'>");
145
146
147         out.println("<tr><td class='kalenteri' style='padding: 4 0 4 0' " +
148                     "colspan='8' align='center'>");
149
150         out.println("<a href='?alkupvm=" + viikkoTaakse + "&" + henkilöTaiRyhmä +
151                     "'>&lt;&lt;</a> ");
152         out.println("  <a href='?alkupvm=" + taakse + "&" +
153                     henkilöTaiRyhmä + "'>&lt;</a> ");
154         out.println(" <a href='?alkupvm=" + eteen + "&" + henkilöTaiRyhmä +
155                     "'>&gt;</a> ");
156         out.println("  <a href='?alkupvm=" + viikkoEteen + "&" +
157                     henkilöTaiRyhmä + "'>&gt;&gt;</a></td></tr><tr>");
158
159
160         /*
161          * Päivien otsikot kalenteriin
162          */
163
164         out.println("<th>klo</th>");
165
166         for (int i = 0; i < 7; i++) {
167           int viikonPäivä = kalenteri.get(Calendar.DAY_OF_WEEK);
168
169           switch (viikonPäivä) {
170             case Calendar.MONDAY:
171                out.println("<th width='100'>Maanantai</th>"); break;
172             case Calendar.TUESDAY:
173                out.println("<th width='100'>Tiistai</th>"); break;
174             case Calendar.WEDNESDAY:
175                out.println("<th width='100'>Keskiviikko</th>"); break;
176             case Calendar.THURSDAY:
177                out.println("<th width='100'>Torstai</th>"); break;
178             case Calendar.FRIDAY:
179                out.println("<th width='100'>Perjantai</th>"); break;
180             case Calendar.SATURDAY:
181                out.println("<th width='100'>Lauantai</th>"); break;
182             case Calendar.SUNDAY:
183                out.println("<th width='100'>Sunnuntai</th>"); break;
184           }
185
186           kalenteri.add(Calendar.DATE, 1);
187         }
188
189         kalenteri.add(Calendar.DATE, -7);
190
191         out.println("</tr>");
192
193
194
195         /*
196          * Varauksien käsittely alkaa
197          */
198
199         Connection con=null;
200         con= createDbConnection(dbDriver,dbServer,dbUser,dbPassword,out);
201         if (con==null) {
202            out.println("</body></html>");
203            return;
204         }
205
206         int num = 0;
207         int aikaisinVaraus = 0;
```

```
208
209
210         Statement stmt = null;
211         ResultSet rs = null;
212
213         try {
214           stmt = con.createStatement();
215
216
217           /*
218            *  Haetaan viikon aikaisimman varauksen alkuaika, jotta voidaan
219            *  halutessa trimmata turhat tyhjät pois aamuista kalenterissa
220            */
221
222           if (henkilo != null) {
223             rs = stmt.executeQuery(
224                 "select min(alkuaika) as aikaisin from varaus where tunnus='" +
225                 henkilo + "' and pvm between to_date('" + viikonAlku +
226                 "', 'DD.MM.YYYY') and to_date('" + viikonLoppu +
227                 "', 'DD.MM.YYYY') group by alkuaika");
228           }
229           else if (ryhma != null) {
230             rs = stmt.executeQuery(
231                 "select min(alkuaika) as aikaisin from varaus where ryhmä='" +
232                 ryhma + "' and pvm between to_date('" + viikonAlku +
233                 "', 'DD.MM.YYYY') and to_date('" + viikonLoppu +
234                 "', 'DD.MM.YYYY') group by alkuaika");
235           }
236
237
238           while (rs.next()) {
239             aikaisinVaraus =
240                Integer.parseInt((rs.getString("aikaisin")).substring(0,2));
241           }
242
243
244           /*
245            *  Kommentoi 'aikaisinVaraus = 0' rivi pois, jos
246            *  haluat, että tila ennen viikon aikaisinta
247            *  varausta trimmataan pois.
248            */
249
250           aikaisinVaraus = 0;
251
252
253         } catch (SQLException ee) {
254             out.println("Tietokantavirhe "+ee.getMessage());
255         } finally {
256           try {
257             if (rs!=null) rs.close();
258             if (stmt!=null) stmt.close();
259
260           } catch(SQLException e) {
261             out.println("An SQL Exception was thrown.");
262           }
263         }
264
265         out.println("<tr><td class='kalenteri' valign='top'>" +
266                     "<table class='ajat' cellpadding='0' cellspacing='0'>");
267
268         for (int i = (0 + aikaisinVaraus); i < 24; i++) {
269           if (i == 23) {
270             out.println("<tr><td class='kalenteri' valign='top' height='" +
271                         ((TUNNIN_KORKEUS)-1) + "'>" + CheckDate.c(i) +
272                         ":00</td></tr>");
273           }
274           else {
275             out.println("<tr><td class='kalenteri' valign='top' height='" +
276                         TUNNIN_KORKEUS + "'>" + CheckDate.c(i) + ":00</td></tr>");
```

```
277         }
278       }
279
280       out.println("</table></td>");
281
282
283
284
285       /*
286        * Käydään seitsemän päivää kalenterin tämänhetkisestä päivämäärästä
287        * lähtien.
288        */
289
290
291       for (int i = 0; i < 7; i++) {  // forin alku
292
293         out.println("<td class='kalenteri' style='background-image: " +
294            "url(http://db.cs.helsinki.fi/u/avtanska/tsoha/img/kale_bg.gif)' " +
295            "valign='top'>");
296
297         String käsiteltäväPäivä =
298             "" + CheckDate.c(kalenteri.get(Calendar.DATE)) + "." +
299             CheckDate.c((kalenteri.get(Calendar.MONTH)+1)) + "." +
300             kalenteri.get(Calendar.YEAR);
301
302
303         try {
304           stmt = con.createStatement();
305           double edellisenLoppu = 0;
306           boolean päivänEnsimmäinen = true;
307
308
309           /*
310            *  Haetaan joko henkilön tai ryhmän varaukset
311            */
312
313           if (henkilo != null) {
314             rs = stmt.executeQuery(
315                 "select * from varaus where tunnus='" + henkilo +
316                 "' and pvm=to_date('" + käsiteltäväPäivä +
317                 "', 'DD.MM.YYYY') order by alkuaika");
318           }
319           else if (ryhma != null) {
320             rs = stmt.executeQuery(
321                 "select * from varaus where ryhmä='" + ryhma +
322                 "' and pvm=to_date('" + käsiteltäväPäivä +
323                 "', 'DD.MM.YYYY') order by alkuaika");
324           }
325
326
327
328           /*
329            *  Käydään yhden päivän kaikki varaukset läpi yksi kerrallaan
330            */
331
332           while(rs.next()) {
333
334             Calendar varausKesto = Calendar.getInstance();
335             String poistoNappi = "";
336             String varausPvm = rs.getString("pvm");
337             String alkuAikaString = rs.getString("alkuaika");
338             String varausAlkaa = "";
339             String varausLoppuu = "";
340             double kesto = 0;
341             double tyhjaTila = 0;
342             double alkuAika = 0;
343
344             alkuAika = Integer.parseInt(alkuAikaString.substring(0,2));
345             kesto = (new Double(rs.getString("kesto"))).doubleValue();
```

```
346
347             if (((rs.getString("alkuaika")).substring(3,5)).equals("30")) {
348               alkuAika = alkuAika + .5;
349             }
350
351
352
353
354             /*
355              *  Lasketaan ja tulostellaan tyhjä tila ennen
356              *  varausta
357              */
358
359             if (päivänEnsimmäinen) {
360               tyhjaTila = alkuAika - aikaisinVaraus;
361             }
362             else {
363               tyhjaTila = alkuAika - edellisenLoppu;
364             }
365
366             edellisenLoppu = alkuAika + kesto;
367
368
369             out.println("<table cellspacing='0' cellpadding='0' " +
370                         "width='100' class='varaus'>");
371
372
373             if (päivänEnsimmäinen && tyhjaTila != 0) {
374               out.println("<tr><td width='100' class='kalenteri' " +
375                  "style='background: url(http://db.cs.helsinki.fi/u/avtanska" +
376                  "/tsoha/img/transparent.gif)' height='" +
377                  ((tyhjaTila*TUNNIN_KORKEUS)-1) + "'></td></tr>");
378             } else if (tyhjaTila != 0) {
379               out.println("<tr><td width='100' class='kalenteri' " +
380                  "style='background: url(http://db.cs.helsinki.fi/u/avtanska" +
381                  "/tsoha/img/transparent.gif)' height='" +
382                  ((tyhjaTila*TUNNIN_KORKEUS)-1) + "'></td></tr>");
383             }
384
385             if ((päivänEnsimmäinen && tyhjaTila != 0) || tyhjaTila != 0) {
386               out.println("<tr><td height='1' style='background: #000000'>" +
387                           "</td></tr>");
388             }
389
390
391             /*
392              * Sitten itse varauksen tulostus
393              */
394
395             out.println("<tr><td valign='top' class='varaus'" +
396                         " height='" + ((kesto*TUNNIN_KORKEUS)-1) + "'>");
397
398             out.println("</td></tr>");
399
400             if (alkuAika + kesto < 24) {
401               out.println("<tr><td height='1' style='background: #000000'>" +
402                           "</td></tr>");
403             }
404
405             out.println("</table>");
406
407             päivänEnsimmäinen = false;
408             num++;
409
410           } // loppu while
411
412         } catch (SQLException ee) {
413               out.println("Tietokantavirhe "+ee.getMessage());
414         } finally {
```

```
415               try {
416                  if (rs!=null) rs.close();
417                  if (stmt!=null) stmt.close();
418
419               } catch(SQLException e) {
420                  out.println("An SQL Exception was thrown.");
421               }
422         }
423
424         kalenteri.add(Calendar.DATE, 1);
425         out.println("</td>");
426
427       } // loppu for
428
429
430       /*
431        *  Suljetaan yhteys
432        */
433
434       try {
435         if (rs!=null) rs.close();
436         if (stmt!=null) stmt.close();
437         con.close();
438       } catch(SQLException e) {
439           out.println("An SQL Exception was thrown.");
440       }
441
442       out.println("</tr><tr><th colspan='8'>Varausten lukumäärä: " +
443                   num + "</th></tr></table></form></body></html>");
444
445     } // loppu else ryhmä ja henkilö ei null
446
447   }
448
449
450
451   private Connection createDbConnection(
452       String dbDriver, String dbServer, String dbUser, String dbPassword,
453       ServletOutputStream out) throws IOException {
454
455       try{
456           Class.forName(dbDriver);
457       } catch (ClassNotFoundException e) {
458             out.println("Couldn't find driver "+dbDriver);
459             return null;
460       }
461       Connection con=null;
462       try {
463          con = DriverManager.getConnection(dbServer,dbUser,dbPassword);
464       } catch (SQLException se) {
465             out.println("Couldn\'t get connection to "+dbServer+
466                         " for "+ dbUser+"<br>");
467             out.println(se.getMessage());
468       }
469       return con;
470   }
471
472 }
```