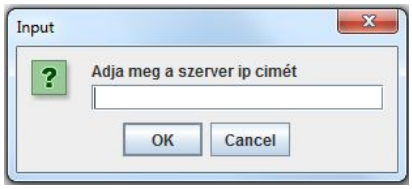
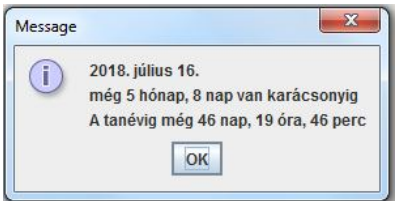
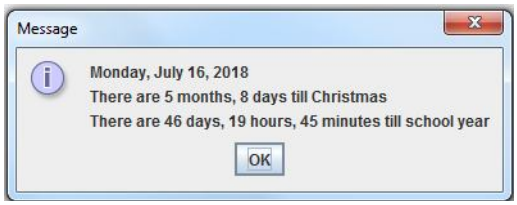
Dátumkezelő kliens-szerver alkalmazás – javított változat

Összeköttetés alapú szolgáltatást alapul véve oldjuk meg, hogy a világ „bármely” táján meg lehessen kérdezni néhány, dátummal kapcsolatos információt.



A kliens kérje be a fix port-számon futó szerver IP-címét, a szerver pedig válaszként különböző (a kliens által megadott) nyelveken közölje az aktuális dátumot, azt, hogy hány hónap és nap van még hátra karácsonyig, illetve hány nap, óra, perc a tanév kezdetéig.





Használja a `JOptionPane` osztály `showInputDialog()` és `showMessageDialog()` metódusát, illetve a JDK_8 `time` csomagjának szolgáltatásait.

Mielőtt nagyon megfeledkezne róla, írjon JUnit tesztet az idők kiszámításának tesztelésére.

Megoldás-részletek

Az előző megoldáshoz képest csak a szerver osztályt módosítjuk, mégpedig úgy, hogy amikor jelentkezik egy kliens, akkor ne kezdje el azonnal kiszolgálni, hanem indítson egy külön szálat a létrejött socket felhasználásával, és azonnal várja a következő kliens jelentkezését. A kiszolgálást pedig ez a szál végzi (vagy több kliens esetén szálak).

Az így módosított osztály kódja:

```
public class DatumosSzerver {

    private static final int PORT = 44444;
    private static boolean online = true;

    public static void main(String[] args) {
        try {
            new DatumosSzerver().start();
        } catch (IOException ex) {
            Logger.getLogger(DatumosSzerver.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

    private void start() throws IOException {
        // A szerver az adott porton várakozik.
        try (ServerSocket serverSocket = new ServerSocket(PORT)) {
            System.out.println("Elindult a szerver");

            while (online) {
                // fogadja a kliens kapcsolódási kérelmét
                Socket socket = serverSocket.accept();

                // elindít egy kiszolgáló szálat
                SzerverSzal szal = new SzerverSzal(socket);
                szal.start();
            }
        }
    }
}

    class SzerverSzal extends Thread {

        private Socket socket;

        SzerverSzal(Socket socket) {
            this.socket = socket;
        }

        @Override
        public void run() {

            LocalDate karacsony = LocalDate.of(2018, 12, 24);
            LocalDateTime tanevkezdoIdoPont
                    = LocalDateTime.of(2018, 9, 1, 8, 00);

            try (BufferedReader input
                    = new BufferedReader(new InputStreamReader(socket.getInputStream()));
                    PrintWriter out
                    = new PrintWriter(socket.getOutputStream(), true)) {

                LocalDate ma = LocalDate.now();

                String nyelv = input.readLine();
                Locale locale = new Locale(nyelv);
```

```
            DateTimeFormatter dtf
                    = DateTimeFormatter.ofLocalizedDate(FormatStyle.FULL);

            dtf = dtf.withLocale(locale);
            String maiKiirtDatum = ma.format(dtf);

            out.println(maiKiirtDatum);

            ResourceBundle bundle
                    = ResourceBundle.getBundle("lokalitasok/DatumBundle",
                            locale);

            Period karacsonyig = Period.between(ma, karacsony);

            String karacsonyigString
                    = String.format(bundle.getString("karacsonyig"),
                            karacsonyig.getMonths(),
                            karacsonyig.getDays());
            out.println(karacsonyigString);

            LocalDateTime mostaniDatumPerc = LocalDateTime.now();

            LocalDateTime tempDateTime
                    = LocalDateTime.from(mostaniDatumPerc);

            long napok
                    = tempDateTime.until(tanevkezdoIdoPont, ChronoUnit.DAYS);
            tempDateTime = tempDateTime.plusDays(napok);

            long orak
                    = tempDateTime.until(tanevkezdoIdoPont, ChronoUnit.HOURS);
            tempDateTime = tempDateTime.plusHours(orak);

            long percek
                    = tempDateTime.until(tanevkezdoIdoPont, ChronoUnit.MINUTES);

            String tanevigString
                    = String.format(bundle.getString("tanevig"),
                            napok, orak, percek);

            out.println(tanevigString);
        } catch (IOException ex) {
            Logger.getLogger(SzerverSzal.class.getName()).log(Level.SEVERE, null, ex);
        }
        finally{
            try {
                socket.close();
            } catch (IOException ex) {
                Logger.getLogger(SzerverSzal.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    }
}
```