Tehtävä 2

Tapahtumaohjelmointi

Tee C#-komentokehoteohjelma (Console Application (.NET Core)). Ohjelmassa on (Program-luokan lisäksi) seuraavat luokat Job, Employee, JobChangedEventArgs, Mediator, Data, ConsoleControl ja Application.

Job

Luokka Job sisältää sisältää automaattiset ominaisuudet Id (kokonaisluku), Title (merkkijono), StartDate (päivämäärä) ja EndDate (päivämäärä).

Employee

Luokka Employee sisältää automaattiset ominaisuudet Id (kokonaisluku), Name (merkkijono) ja Jobs (Job-olioiden lista).

JobChangedEventArgs

Luokka JobChangedEventArgs periytyy luokasta System.EventArgs sisältäen automaattisen ominaisuuden Job (tyyppiä Job).

Mediator

Suunnittelumallit (design patterns) ovat olio-ohjelmoinnissa käytettäviä koodimalleja, jotka ovat käytännön soveltamisen kautta hyviksi havaittuja. Suunnittelumallit ovat olio-ohjelmointikielestä riippumattomia abstrakteja malleja. Tutustu halutessasi seuraaviin suunnittelumalleihin (- ei kuitenkaan välttämätöntä tehtävän tekemiseksi):

- Singleton eli ainokainen (https://en.wikipedia.org/wiki/Singleton_pattern) ja
- Mediator eli välittäjä (https://en.wikipedia.org/wiki/Mediator_pattern).

Suljettu luokka Mediator toteuttaa molemmat edellä mainituista suunnittelumalleista sisältäen seuraavat elementit:

- staattinen Mediator-tyyppinen private muuttuja nimeltä instance, jonka arvoksi asetetaan määrittelyn yhteydessä Mediator-olio.
- staattinen ominaisuus Instance, jossa on vain get-metodi palauttaen staattisen muuttujan instance arvon.
- piilotettu konstruktori eli private-näkyvyydellä varustettu konstruktori, jossa ei ole toimintaa. Tämän tarkoitus on estää olioiden tekeminen new-komennolla muualla kuin itse luokassa (vrt. muuttujan instance alustaminen).
- tapahtuma JobChanged, jonka tyyppinä on geneerinen delegaattityyppi System.EventHandler<JobChangedEventArgs>.
- metodi OnJobChanged, jossa on parametrit sender (tyyppiä object) ja job (tyyppiä Job). Metodin toimintana on suorittaa tapahtumassa JobChanged olevat mahdolliset tapahtumakäsittelijät. Toteutus on seuraava:
    - paikalliseen muuttujaan jobChangeDelegate sijoitetaan arvoksi tapahtuman JobChanged arvo tyypitettynä as-operaattorilla System.EventHandler<JobChangedEventArgs>-tyyppiseksi (- näin muuttujan arvoksi tulee joko tapahtuman arvo tai null).

- jos edellä tehdyn muuttujan arvo on jotain muuta kuin null, kutsutaan muuttujan arvona olevaa metodia argumentein sender (metodin parametri) ja uusi JobChangedEventArgs-olio, jonka Job-ominaisuuden arvoksi parametrin job arvo.

Data

Luokka Data sisältää staattiset muuttujat jobs ja employees, jotka simuloivat ohjelman käyttämää dataa. Normaalisti sovellus tuottaisi tämän kaltaiset oliot tietokannan datan pohjalta. Tässä muuttujien määrittelyt valmiina (- Huom! Jos kopiot koodin pdf-tiedostosta, kirjoita -merkit Visual Studiossa uudelleen , sillä ovat pdf:ssä eri merkkejä, vaikka näyttävät samalta.):

```
public static List<Job> jobs = new List<Job> {
    new Job { Id = 1, Title = "Siiven C luokkien ylläpito",
              StartDate = DateTime.Today, EndDate = DateTime.Today.AddDays(1) },
    new Job { Id = 2, Title = "Parkkialueiden maalaus",
              StartDate = DateTime.Today.AddDays(-5), EndDate = DateTime.Today.AddDays(5) },
    new Job { Id = 3, Title = "Puiden istutus",
              StartDate = DateTime.Today.AddDays(4), EndDate = DateTime.Today.AddDays(10) },
    new Job { Id = 4, Title = "Väliseinän rakentaminen",
              StartDate = DateTime.Today, EndDate = DateTime.Today.AddDays(15) }
};

public static List<Employee> employees = new List<Employee> {
    new Employee { Id = 1, Name = "Matti Mainio",
                   Jobs = new List<Job> {jobs[0],jobs[1],jobs[2],jobs[3]}},
    new Employee { Id = 2, Name = "Jussi Juonio",
                   Jobs = new List<Job> {jobs[2],jobs[3]}},
    new Employee { Id = 3, Name = "Maija-Liisa Lahtinen",
                   Jobs = new List<Job> {jobs[1],jobs[3]}},
    new Employee { Id = 4, Name = "Anneli Aalto",
                   Jobs = new List<Job> {jobs[0],jobs[1],jobs[3]}},
    new Employee { Id = 5, Name = "Pentti Penttilä",
                   Jobs = new List<Job> {jobs[2]}},
    new Employee { Id = 6, Name = "Sakari Ruoho",
                   Jobs = new List<Job> {jobs[0],jobs[3]}}
};
```

ConsoleControl

Luokka ConsoleControl sisältää seuraavat elementit:

- automaattiset ominaisuudet Items (merkkijonolista), Column, Row, Width, Height (kokonaislukuja), BackColor ja TextColor (ConsoleColor-luettelotyypin tyyppisiä).
- konstruktori, jossa neljä kokonaislukuparametria col, row, width ja height. Toimintana parametrien arvot sijoitetaan vastaavien ominaisuuksien (Column, Row, Width, Height) arvoiksi. Ominaisuuteen BackColor sijoitetaan arvoksi System.Console.BackgroundColor ja ominaisuuteen TextColor arvoksi System.Console.ForegroundColor. Ominaisuuteen Items sijoitetaan arvoksi null.
- metodi Clear, joka pyyhkii olion määrittelemän alueen kirjoittamalla välilyöntejä koko alueelle.
    - Olion määrittelemä alue tarkoittaa ominaisuuksien Column, Row, Width ja Height rajaamaa aluetta komentokehoteikkunassa. Column ja Row määrittelevät alueen vasemman ylänurkan koordinaatit (sarakenumero ja rivinumero). Width ja Height puolestaan määrittelevät alueen leveyden merkkeinä ja korkeuden riveinä.
    - Vihje: ota talteen kursorin sijainti ennen puhdistuksen alkua ja palauta kursori takaisin puhdistuksen jälkeen. Kursorin nykyisen sijainnin saa selville System.Console-luokan staattisista ominaisuuksista CursorLeft (nykyinen sarake) ja CursorTop (nykyinen rivi).

- metodi Draw, joka piirtää määrittelyn alueen komentokehoteikkunaan. Käytä piirtämisessä väreinä olion ominaisuuksissa TextColor ja BackColor olevia arvoja.
    - Alueen piirtäminen tarkoittaa ominaisuuden Items sisältönä olevan merkkijonolistan merkkijonojen kirjoittamisen ominaisuuksien Column, Row, Width ja Height rajaamalle alueelle (ks. metodi Clear). Listan merkkijonot kirjoitetaan listan mukaisessa järjestyksessä kukin omalle rivilleen. Jos yksittäisen merkkijonon pituus on pienempi kuin alueen leveys (Width), kirjoitetaan rivin loppuosaan välilyöntimerkkejä (= kaikki kirjoitettavat rivit ovat siis vähintään ominaisuuden Width-pituisia).
    - Vihje 1: ota talteen sekä kursorin sijainti että kirjoittamisen värit ennen piirtämisen aloittamista, jotta voit palauttaa ne takaisin piirtämisen lopuksi. Nykyisen kirjoittamisen värit saadaan System.Console-luokan staattisista ominaisuuksista ForegroundColor ja BackgroundColor.
    - Vihje 2: merkkijono-olion metodi PadRight.

Application

Luokka Application on staattinen ja se toimii sovelluksen “ytimenä”. Luokka sisältää seuraavat staattiset elementit (kaikki elementit ovat yksityisiä (private) ellei toisin mainita):

- ConsoleControl-tyyppiset muuttujat JobMenu, JobDetails ja JobEmployees.
- metodi BindMenuData, jolla on List<Job>-tyyppinen parametri ja jossa toimintana populoidaan muuttujan JobMenu ominaisuus Items. Populointi = Lisätään jokaisesta parametrin sisältämän listan Job-oliosta merkkijono muodossa Id Title (esim. ”3 Puiden istutus”) ominaisuuteen Items.
- metodi BindDetailsData, jolla on Job-tyyppinen parametri.
    - Jos staattisen muuttujan JobDetails ominaisuuden Items arvo on null, luodaan sen arvoksi merkkijonojen lista. Muuten tyhjennetään arvona jo oleva merkkijonojen lista (- listalla on metodi Clear, jolla kaikki listan oliot voidaan poistaa kerralla).
    - Populoidaan Items-ominaisuus seuraavilla Job-olion ominaisuuksista saatavilla merkkijonoilla (tässä job on parametrin nimi) (populointi = lisätään listalle merkkijonoja):

```
"TYÖN TIEDOT"
$"Id: {job.Id}"
$"Nimi: {job.Title}"
$"Alkaa: {job.StartDate.ToShortDateString()}"
$"Loppuu: {job.EndDate.ToShortDateString()}"
```

- metodi BindEmployeesData, jolla on Job-tyyppinen parametri.
    - Samanlainen Items-ominaisuuden käsittely kuin edellä. Jos staattisen muuttujan JobEmployees ominaisuudessa Items ei vielä ole merkkijonolistaoliota, sijoitetaan arvoksi uusi, muuten tyhjennetään olemassa oleva lista.
    - Populoidaan Items-ominaisuus merkkijonolla ””TYÖN TEKIJÄT” ja parametrin Job-olioon liittyvien Employee-olioiden Name-ominaisuuksien arvoilla (eli työhön liittyvien työntekijöiden nimillä). (Ks. esimerkkisuoritus tehtävän lopussa.)
        - Job-olioon liittyvät Employee-oliot saat käymällä läpi muuttujan Data.employees oliot ja tutkimalla niistä, onko oliolle määritelty Jobs-ominaisuuteen ko. Job-olio (eli onko parametrin Job-olion Id sama kuin Jobs-ominaisuuden Job-olion Id).
- metodi Initialize, jossa

    - edellä määriteltyihin staattisiin muuttujiin JobMenu, JobDetails ja JobEmployees sijoitetaan arvoiksi ConsoleControl-oliot.
        - ConsoleControl-olioiden Column, Row, Width ja Height-ominaisuudet määrittelevät olion piirtoalueen komentokehoteikkunassa (vasen yläkulma eli sarake ja rivi sekä leveys ja korkeus).
        - anna olioiden konstruktoreille argumentit niin, että
            - valikko (JobMenu) tulostuu ikkunassa sarakkeeseen 1 riville 2 ja valikon leveys on puolet ikkunan leveydestä (System.Console.WindowWidth) vähennettynä yhdellä ja korkeus on staattisen muuttujan Data.jobs sisältämien Job-olioiden määrä.
            - työn tiedot (JobDetails) tulostuu valikon oikealle puolelle. Sarake on ikkunanleveys/2 + 1 ja rivi on 2. Leveys on puolet ikkunan leveydestä (System.Console.WindowWidth) vähennettynä yhdellä ja korkeus on 5.
            - työn tekijät (JobEmployees) tulostuu työn tietojen alle. Sarake on sama kuin työn tiedoille ja rivi on JobDetails-muuttujan Height-ominaisuuden arvo lisättynä kolmella. Leveys on sama kuin työn tiedoissa. Korkeus on ikkunan korkeus (System.Console.WindowHeigth) vähennettynä JobDetails-muuttujan Height-ominaisuuden arvolla ja yhdellä.
            - Katso tehtävän lopusta mallisuorituksen tuottama tulos, jotta saat käsityksen tavoitelluista sijainneista.
        - voit konstruktorien kutsun yhteydessä (tai erikseen niiden jälkeen) asettaa olioiden TextColor ja BackColor-ominaisuuksiin haluamasi värit.
            - Mallisuorituksessa on käytetty seuraavia värejä: BackColor on kaikissa ConsoleColor.Gray ja tekstin värit valikossa ConsoleColor.DarkBlue, työn tiedoissa ConsoleColor.DarkGreen ja työn tekijöissä ConsoleColor.DarkRed.
    - suoritetaan metodi BindMenuData antaen argumentiksi luokan Data staattinen muuttuja jobs.
    - asetetaan Mediator-olion JobChanged-tapahtumalle tapahtumakäsittelijä, jossa kutsutaan metodit BindDetailsData ja BindEmployeesData antaen molemmissa argumentiksi tapahtumakäsittelijän saaman JobChangedEventArgs-olion Job-ominaisuuden arvo.
        - Vihje: tapahtumakäsittelijän (metodin) voi toteuttaa lambda-lausekkeena tai normaalisti omana nimettynä metodina.
- metodi MenuSelectionChanged, jossa on kokonaislukutyyppinen parametri ilmaisten valitun työn Id:n arvon. Toimintana on seuraava:
    - etsitään staattisen muuttujan Data.jobs sisältämästä listasta Job-olio, jolla on parametrin arvona annettu Id-ominaisuuden arvo.
    - jos ja kun oikea Job-olio on löydetty, synnytetään Mediator-oliolle tapahtuma JobChanged kutsumalla metodi Mediator.Instance.OnJobChanged ja antamalla argumenteiksi JobMenu-muuttujan olio sekä edellä löydetty Job-olio.
    - tulostetaan työn tiedot ja tekijöiden nimet kutsumalla metodia Draw muuttujien JobDetails ja JobEmployees sisältämille oliolle.
- julkinen (public) metodi Run, jonka toiminta on seuraava:
    - kutsutaan metodi Initialize
    - piirretään valikko kutsumalla muuttujan JobMenu sisältämän olion metodi Draw.
    - toistetaan seuraavaa rutiinia, kunnes käyttäjä syöttää nollan.

- asetetaan kursori ikkunan vasempaan ylänurkkaan (= System.Console.SetCursorPosition(0,0)).
- pyydetään käyttäjältä jokin valikon esittämien töiden id-arvoista.
    - Jos syöte ei ole kelvollinen (eli muu kuin luku väliltä [0 - JobMenu.Items.Count]), asetetaan kursori ikkunan vasempaan ylänurkkaan ja tulostetaan virheteksti sekä pyydetään kuittaus.
- Jos edellä syöte oli kelvollinen, esitetään saadun arvon perusteella vastaavan Job-olion tiedot ja siihen liittyvät tekijät seuraavasti:
    - Kutsutaan metodi MenuSelectionChanged antaen argumentiksi edellä syötetty id-arvo.

Program

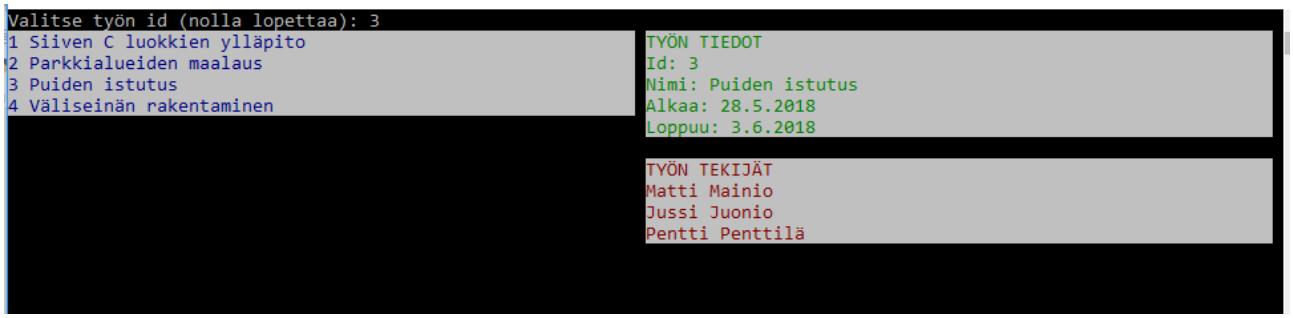
Program-luokan Main-metodissa kutsutaan metodia Application.Run.

Esimerkki ohjelman toiminnasta

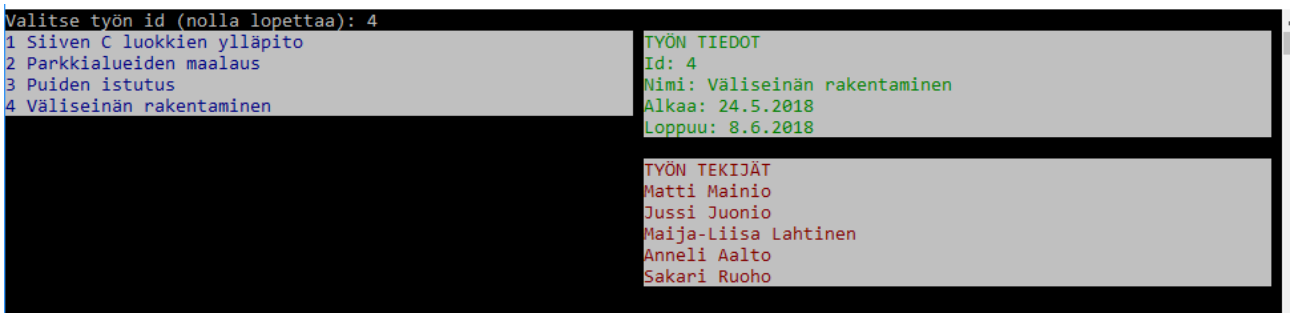
Aloitus:

Syötetään virheellinen arvo:

Painetaan Enter ja syötetään ensin arvo 3



ja sitten arvo 4



Työn tietojen ja tekijöiden osuus muuttuu ”automaattisesti” työn id:n syöttämisen yhteydessä. Työn id:n syöttäminen aiheuttaa tapahtuman JobChanged, jonka tapahtumakäsittelijä sisältää tietojen ja tekijöiden päivittämisen.

Mediator-suunnittelumallin avulla toteutuksessa on saavutettu kolmen eri ”komponentin” välille tapahtumakytkentä ilman, että komponentit tietävät mitään toisistaan.