Rajapinnat lopputyö

Tehtävän anto

Ohjelma, joka Scrooge tarvitsee työkalun analysoidakseen bitcoinin markkina-arvoa haluamallaan ajanjaksolla. Ohjelma voidaan toteuttaa, millä tahansa tekniikalla.

Lisätietoja Scrooge antoi, että ohjelman pitäisi kerätä seuraavat tiedot eri ajanjaksoilta:

- alkamis- että lopetuspäivät tulee sisältyä tarkkaan ajanjaksoon.
- Päivähinta haetaan aina, jolloin hinnan määritys aika on lähimpänä 00:00 UTC aikaan
- Käyttäjän tulee voida syöttää alkamis- ja lopetuspäivämäärät (esimerkiksi syöttökentillä)

Päätin valita Rajapinnat lopputyöksi T2 esimerkin rajapinnan ja toteutin sen annettujen ohjeiden mukaisesti. Tein tehtävät A-D.

Layoutin suunnittelu

Aluksi aloitin suunnittelemaan layouttia. Halusin tehdä layoutista sellaisen, että se oli mahdollisimman yksinkertaisen näköinen ja kaikki elementit olisi sijoitettu järkevästi. Halusin tehdä alkamis- ja lopetuspäivämäärä lisäyksen tekstilaatikoilla. Toiveeni oli, että voisi jotenkin valita, missä järjestyksessä suorittaisin tehtävät ja päätin haluta lisätä comboboxin eli monivalintalaatikon, jossa olisi kaikki tehtävät valittavissa. Lisäksi halusin lisätä napin, jolla voisi suorittaa ohjelman ja ohjelmasta tullut suoritus tulisi näkyviin datagridview elementtiin.

Tehtävän pohjan luominen

Halusin tehdä ohjelman rakenteesta selkeän. Tehtävän annossa pyydettiin noudattamaan **MVC** mallia, eli (**M**odel, **V**iew, **C**ontroller).

Model:

- Sisältää BitcoinData luokan, johon tallennan tiedot CoinGecko apista.

View:

- Sisältää Formin tiedostot eli sen, mikä näkyy käyttäjälle

Controller:

- **AllButtonsController**, jota voin kutsua Formissa, jotta ei tarvitse jokaista luokkaa kutsua erikseen, tämä mielestäni selkensi ohjelmaa paljon. Rakenne on jokaisen luokan eri osion hakeminen ja palauttaminen.
- **ApiController**, joka hoitaa apin käsittelyyn kaiken tarvittavan homman, kuten hakee tiedot, ottaa vain keskiyön datan ja kaikkea muuta.
- **DateTimeParse**, joka hoitaa unixtimen muuttamisen normaaliin aika muotoon ja toisten päin. Lisäksi tämä luokka käsittelee kaikki päiviin – aikaan liittyvät asiat.
- **FormController**, joka hoitaa kaikki formiin liittyvät jutut, esimerkiksi nappien toiminta, syötteiden tarkistustuksen.
- Tämän lisäksi loin Controller kansioon kansion Task A-D, johon laitoin kaikki tehtävien ratkaisut selkeästi jaoteltuna

Tehtävä A – Toteutus

Tehtävän anto: Minä päivänä tietyllä päivämäärävälillä oli halvin ja korkein hinta?

Aluksi mietin, että miten saisin tehtyä tehtävän yksinkertaisesti. Keksin, että jos haluan halvimman ja korkeimman hinnan niin tällöin hinnat pitäisi järjestää listan suurimmasta pienimpään hinnan mukaan, jotta saisin korkeimman hinnan ja halvimmalle hinnalle pienimmästä suurimpaan.

Käytin tehtävän toteutuksessa Orderby ja OrderByDescending Linq syntaksin osaa, jotta sain järjestettyä listan järkevästi.

Järjestämisen jälkeen, piti ottaa valitun itemin ensimmäinen osa [0] ja muuttaa se DateTimeksi ja toisen osan [1] muuttaen sen doubleksi, koska kyseessä oli valuutta. Lopuksi lisäsin kummastakin saadut tiedot datagrid elementtiin.

Tehtävä B – Toteutus

Tehtävän anto: Minä päivänä tietyllä päivämäärävälillä kaupankäyntivolyymi oli pienin ja suurin?

Tehtävän B:n toteutus toimii käytännössä samalla tavalla kuin Tehtävä A:n toiminta. Ainut eroavaisuus on siinä, että tässä käsitellään Volume listaa, kuin taas A tehtävässä Prices listaa.

Tehtävä C – Toteutus

Tehtävän anto: Kuinka monta päivää on pisin lasku- (laskeva) ja nousutrendi (nouseva) tietyllä aikavälillä?

Lähdin toteuttamaan tehtävää aluksi niin, että loin methodit Pisinlasku ja Pisinnousu, koska halusin erotella tehtävän kahteen osaan, jolloin niitä olisi helpointa käsitellä.

Seuraavaksi piti järjestää listä unixtimestampin mukaan, jotta saisin oikeat arvot lopputuloksessa. Kummassakin methodissa käyn läpi prices listaa, ottaen sieltä aina foreach silmukkaa käyttäen ensimmäisen itemin. Tämän jälkeen loin if-else kyselyn, jotta saisin tehtävässä halutut tiedot. Lopuksi lisäsin saadut tiedot datagridview elementtiin.

Tehtävä D - Toteutus

Tehtävän anto: Sovelluksen tulisi pystyä kertomaan tietyllä aikavälillä paras päivä bitcoinin ostamiseen ja paras päivä myydä ostettu bitcoin voittojen maksimoimiseksi ja päinvastoin, esim. myy ensin ja osta takaisin myöhemmin.

Ennen kuin lähdin toteuttamaan tehtävää niin halusin tässäkin tehtävässä jakaa tehtävän kahteen osaan. Nimesin tehtävän methodit (GetBestDateToSellOrBuy ja GetBestDateToBuyorSell).

GetBestDateToSellOrBuy

Tässä metodissa otan parametrina BitcoinData-olion ja otin sen prices-listan selvittääkseni, milloin Bitcoinin hinta on ollut alimmillaan eli paras hetki ostaa. Aluksi järjestin listan nousevaan järjestykseen ja otin sieltä ensimmäisen itemin, joka sisältää halvimman hinnan sekä siihen kuuluvan aikaleiman. Muutin aikaleiman DateTime-muotoon ja tallensin sen myöhempää käyttöä varten.

Seuraavaksi kävin alkuperäisen listan läpi foreach-silmukalla ja rakensin if-rakenteen, jonka avulla varmistin esimerkiksi, että myyntitapahtuma tapahtuu ennen ostoa. Lisäksi laskin jokaisesta tilanteesta mahdollisen voiton ja säilytin parhaan löydetyn tuloksen. Lopuksi lisäsin tiedot DataGridView elementtiin.

GetBestDateToBuyorSell

Tässä metodissa otan parametrina BitcoinData-olion ja otin sen prices-listan selvittääkseni, milloin Bitcoinin hinta on ollut korkeimmillaan eli paras hetki myydä. Aluksi järjestin listan laskevaan järjestykseen ja otin sieltä ensimmäisen itemin, joka sisältää korkeimman hinnan sekä siihen kuuluvan aikaleiman. Muutin aikaleiman DateTime-muotoon ja tallensin sen myöhempää käyttöä varten.

Seuraavaksi kävin alkuperäisen listan läpi foreach-silmukalla ja rakensin if-rakenteen, jonka avulla varmistin esimerkiksi, että ostoajankohta tapahtuu ennen myyntiä. Lisäksi laskin

jokaisesta tilanteesta mahdollisen voiton ja säilytin parhaan löydetyn tuloksen. Lopuksi lisäsin tiedot DataGridView elementtiin.

Loppuraportti

Minusta onnistuin tehtävän toteutuksessa todella hyvin. Sain kaikki toiminnallisuudet toimimaan halutulla tavalla ja luomaan ohjelman rakenteen mahdollisimman yksinkertaisesti ja tehokkaasti, niin että sitä on helppo myöhemmin muokata tai lisätä ominaisuuksia ilman, että koko rakenne menisi rikki. Antaisin itselleni arvosanaksi 5, koska ohjelma on luotu laadukkaaksi, selkeäksi ja helposti muutettavaksi myös aloittelija tasolle.