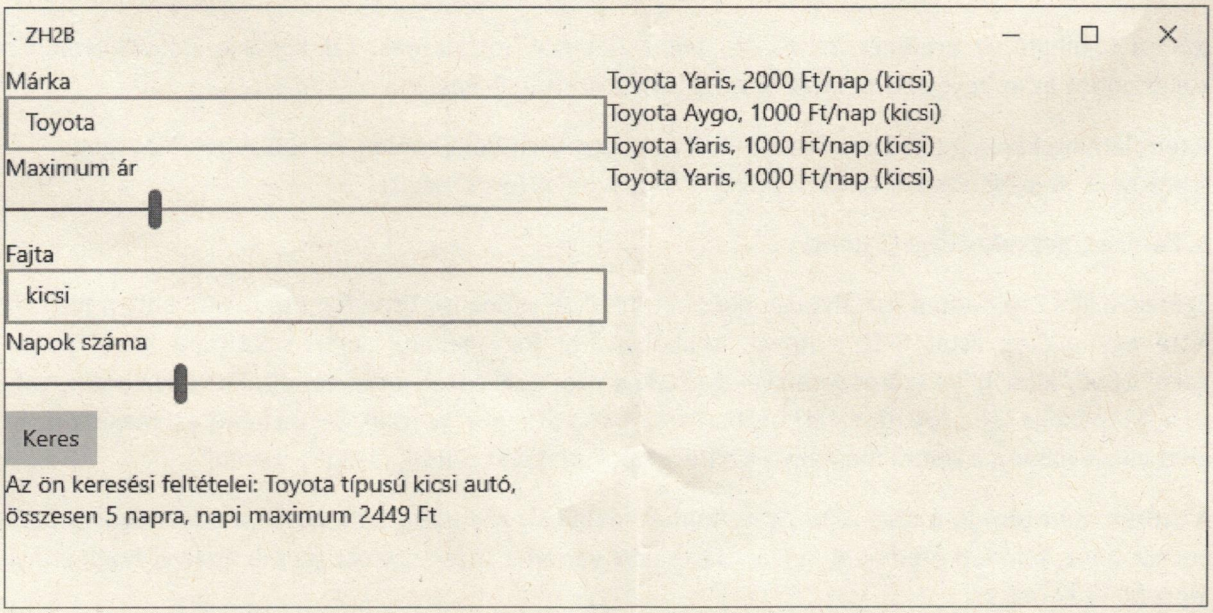
Eseményvezérelt és Vizuális Programozás ZH 2 – 2022. ősz – B csoport

A feladat egy autókölcsönző keresőjének elkészítése UWP nyelven, melynek kinézete közelítőleg megegyezik az alábbi ábrán láthatóval:



A felület két részre van osztva: baloldalt láthatóak a szűrőfeltételek, jobboldalt pedig a találatok, amiket akkor listáz ki a program, amikor a felhasználó rákattint a keresés gombra. A keres gomb alatt pedig egy összefoglaló látható a keresésről.

A megoldáshoz a következő feladatokat végezze el:

1. Megoldás vázának létrehozása (8 pont)

A megoldáshoz a következő osztályokat hozza létre/egészítse ki:

- *Car:* tartalmazza az autók tulajdonságait 3 property formájában: *Name* (string), *Price* (int), *Type* (string). Ezeket az értékeket a konstruktorban várja.
- *DataModel*: tartalmaz egy *Cars* kollekciót, ami tárolja a jobb oldalt megjelenítendő találatokat, valamint itt találhatóak a szűrőfeltételek. Az adatkötés miatt implementálja az INotifyPropertyChanged interfészt. A szűrőfeltételeket kell majd kötni a felületen lévő elemekhez, amik a következőek legyenek: *Brand* (string), *MaxPrice* (int), *SelectedType* (string), *Days* (int). Ezen kívül legyen egy *Summary* (string) változó, mely a következő formában adja vissza a keresés összefoglalóját az alábbi példa szerint: "Az ön keresési feltételei: Toyota típusú kicsi autó, összesen 5 napra, napi maximum 2449 Ft". Konstruktorban adjon hozzá néhány autót a *Cars* kollekcióhoz
- *MainPage:* Tartalmazzon egy DataModel típusú *Model* változót, amit a konstruktorban példányosítson!

2. Felhasználói felület elkészítése (8 pont)

A felhasználó felület alapja egy két oszlopból álló grid. A bal oldali oszlopban egymás alatt vannak a szűrőfeltételek a képen látható módon. A feltételeknél minden egyes szerkeszthető értékhez legyen odaírva, hogy mit szerkesztünk (például „Márka"), majd utána következzen az érték szerkesztéséhez való Control. A string típusú értékeket TextBox segítségével lehessen szerkeszteni, míg a számokat egy

csúszkával. A csúszkákhoz tartozzon minimális és maximális érték! Az értékek legyenek kötve a *DataModel* megfelelő változóihoz, figyeljen az adatkötés módjára és irányára! A szűrőfeltételek alatt legyen egy „Keres" feliratú gomb, az alatt pedig a keresés összegzése (a *DataModel.Summary* változóhoz van kötve) mely frissül, amikor egy-egy keresési feltételt szerkesztünk.

Jobbra található az eredmények listája, amin az autók találhatóak. Ezt kösse a *DataModel.Cars* kollekcióhoz és az egyes elemekben az autók nevei szerepeljenek.

A fent látható kép segít az elrendezés módjában.(tipp: TextBlock esetén, ha többsoros szöveget szeretnénk megjeleníteni a TextWrapping="WrapWholeWords" segít).

3. Keresés megvalósítása (7 pont)

Egészítse ki a DataModel osztályt egy paraméternélküli aszinkron *Search* metódussal! Ebben hozzon létre egy *allCars* listát 8-10 autóval, amiből szűrni fog (például „allCars.Add(new Car("Toyota Yaris",2000,"kicsi"))"! A szűrés során válogassa ki azokat az autókat, amelyeknek a nevében benne van az autó márkája (pl. „Toyota Yaris" névben szerepel a „Toyota" márka), az ára a szűrési maximum ár alatt van, valamint a típusa megegyezik a beírt típussal (lehet „kicsi", „nagy", „kombi").

A szűrés után törölje a *Cars* aktuális tartalmát majd egy ciklusban adja hozzá a találatokat a *Cars* kollekcióhoz, minden eredmény hozzáadása előtt várjon 1 másodpercet (szimbolizálva, hogy ennyi ideig tart a keresés).

Készítsen egy *SearchCommand* osztályt, mely implementálja az ICommand interfészt! A *SeachCommand* a konstruktorban vár egy DataModel példányt amit elment egy privát tagváltozóban. Az Execute függvény szintén legyen aszinkron és hívja meg a fent létrehozott *Search* metódust amennyiben a szűréshez használt *Brand* és *SelectedType* be van állítva (a validálásnál használja ugyanezt a tagváltozót).

A felületen található gomb Command tulajdonságát kösse egy *SearchCommand* példányhoz, amit hozzon létre a MainPage-ben és inicializálja a konstruktorában!

4. Lista szépítése (2 pont)

Egészítse ki a *Car* osztályt egy *Description* (string) property-vel, mely visszaadja az autó leírását az alábbi formában: "Toyota Yaris, 2000 Ft/nap (kicsi)". Ezt a tagváltozót jelenítse meg a találati listában!

Egyéb tudnivalók:

- A megoldáshoz használjon Blank App (Universal Windows) projekt típust, C# nyelven.
- A futtatáshoz ellenőrizze, hogy a Configuration Managerben (processzor típus legördülő listája) be van X-elve a projektnél a „Deploy".
- Az ICommand interfész a System.Windows.Input névtérben található. A nyomógombot a Command attribútumával használja, ne a Click eseménykezelővel!

A feladat leadása határidőre az első ZH-hoz hasonlóan történik, github pull request formájában, melynek szövegébe helyezzen el egy screenshotot arról az állapotról ameddig eljutott. Figyeljen rá, hogy a leadott verzió forduljon! Rendelje hozzá a laborvezetőjét reviewerként. A ZH befejezési ideje az utolsó commit és pull request időpontjára vonatkozik, a végső verziónak fordulnia kell. Későbbi leadásra percenként 2 pont levonás jár. Amennyiben nincs kép az alkalmazásról, vagy nem fordul, 5-5 pont levonás jár.