Eötvös Loránd Tudományegyetem
Informatikai Kar

Eseményvezérelt alkalmazások fejlesztése II

13. előadás

Az adatkezelés eszközei (ADO.NET)

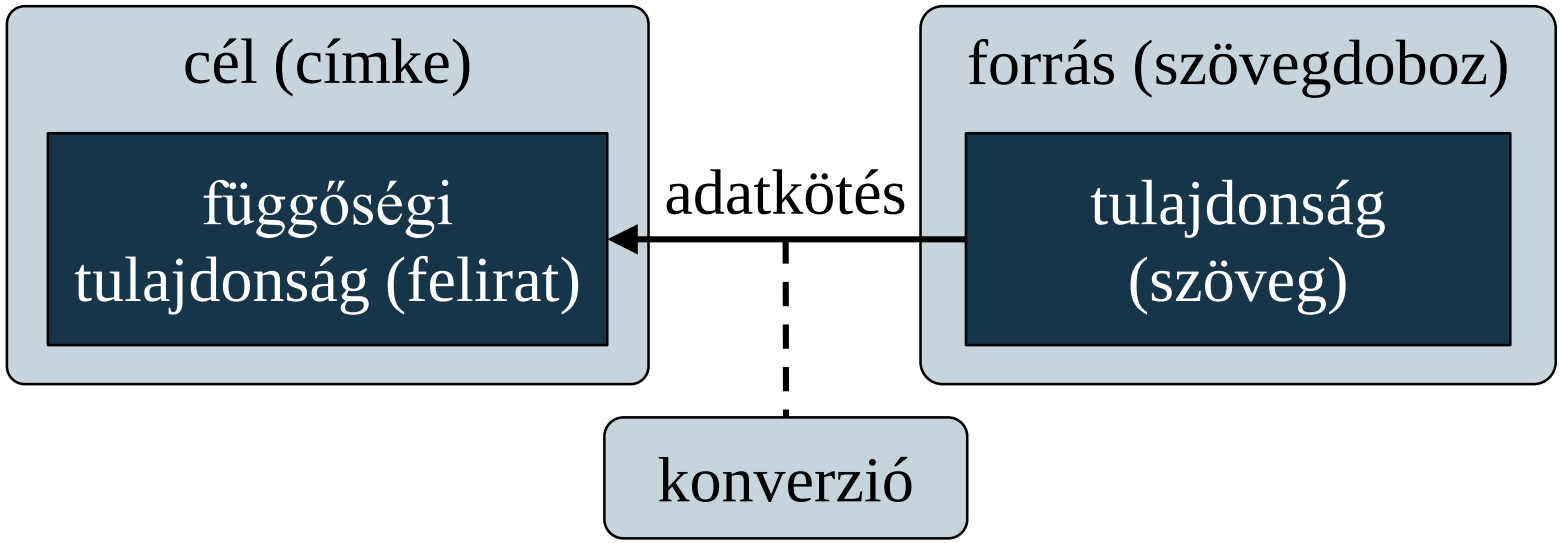
Giachetta Roberto

A jegyzet az ELTE Informatikai Karának
2014. évi Jegyzetpályázatának támogatásával készült

Az adatkezelés eszközei

Adatkonverzió

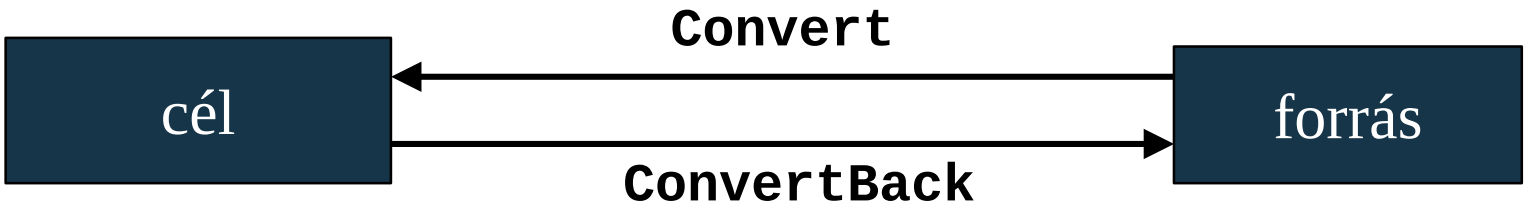
- Az adatkötés során lehetőségünk van átalakítani (konvertálni) az adatot a megjelenítés és a kötött tartalom között
  - vannak alapértelmezett átalakítások (pl. szöveg/szám)
  - alkalmazhatunk egyedi konverziót (az **`IValueConverter`** interfész megvalósításával)



Az adatkezelés eszközei

Adatkonverzió

- Az **`IValueConverter`** interfész biztosítja a **`Convert`** és **`ConvertBack`** műveleteket, amelyek elvégzik a transzformációt
  - tetszőleges típusok között végezhető konverzió
  - figyelembe vehetjük a nyelvi környezetet (**`CultureInfo`**)



- A konverziót a kötésnél adjuk meg, általában erőforrásból betöltve:

```
{Binding Path=…, Converter=…,
         ConverterParameter=…, ConverterCulture=…}
```

Az adatkezelés eszközei

Adatkonverzió

- Pl.:

```
class StringToIntConverter : IValueConverter
   // egyszerű szám-szöveg átalakító
{
   public object Convert(object value, …){
      return value.ToString();
   } // átalakítás szöveggé

   public object ConvertBack(object value, …){
      return Convert.ToInt32(value);
   }
   // átalakítás számmá
}
```

Az adatkezelés eszközei

Adatkonverzió

- Pl.:

```
<Window … xmlns:local="clr-namespace:MyApp">
   <Window.Resources>
      <local:StringToIntConverter
         x:Key="converter" />
      <!- az átalakító, mint erőforrás -->
   </Window.Resources>
   …
   <TextBox Text="{Binding Path=…,
      Converter={StaticResource converter}
      }" />
   <!-- szövegdoboz, amely az adatkötéshez
        felhasználja az átalakítót -->
   …
```

Az adatkezelés eszközei

Adatkonverzió hibakezelése

- Az átalakítás során hiba léphet fel (pl. a megadott szöveg nem konvertálható számmá), amelyet megfelelően kell kezelnünk
  - a konvertáláskor nem keletkezhet kivétel
  - amennyiben a cél értékét nem tudjuk létrehozni (a **`Convert`** műveletben), akkor jelezzük, hogy nem kell végrehajtani a kötést (**`Binding.DoNothing`**)
  - amennyiben a forrás tulajdonságot nem tudjuk beállítani (a **`ConvertBack`** műveletben), akkor visszaadjuk a beállítatlan függőségi értéket (**`DependencyProperty.UnsetValue`**)
  - a beállítási hiba azonnal jelentkezik a felületen is (alapértelmezetten piros keretben)

Az adatkezelés eszközei

Adatkonverzió hibakezelése

- Pl.:

```
class StringToIntConverter : IValueConverter
   // egyszerű szám-szöveg átalakító
{
   …
   public object ConvertBack(object value, …){
      try {
         return Convert.ToInt32(value);
      } catch { // elfogjuk a kivételt
         return DependencyProperty.UnsetValue;
      } // jelezzük a sikertelen beállítást
   } // átalakítás számmá
}
```

Az adatkezelés eszközei

Ellenőrzések adatkonverzióval

- A hibakezeléssel egybekötött átalakító használható ellenőrzések végrehajtására is
  - nem is szükséges konvertálnia a tartalmat, csupán ellenőrzi az adat meglétét, formátumát
- Pl.:

```
class EmailCheckConverter : IValueConverter
   // e-mail formátum ellenőrző átalakító
{
   public object Convert (object value, …){
      return value;
   } // nem végzünk semmilyen átalakítást
```

Az adatkezelés eszközei

Ellenőrzések adatkonverzióval

```
public object ConvertBack(object value, …){
   if (value == null ||
       !Regex.IsMatch(value.ToString(),
           @"^([0-9a-zA-Z]([-.\w]*[0-9a-zA-
             Z])*@([0-9a-zA-Z][-\w]*[0-9a-zA-
             Z]\.)+[a-zA-Z]{2,9})$")){
      // ha nem egyezik az e-mail formátum
      // reguláris kifejezésével
      return DependecyObject.UnsetValue;
      // akkor jelezzük a hibát
   }
   return value;
   // különben nem csinálunk semmit
}
}
```

Az adatkezelés eszközei

Példa

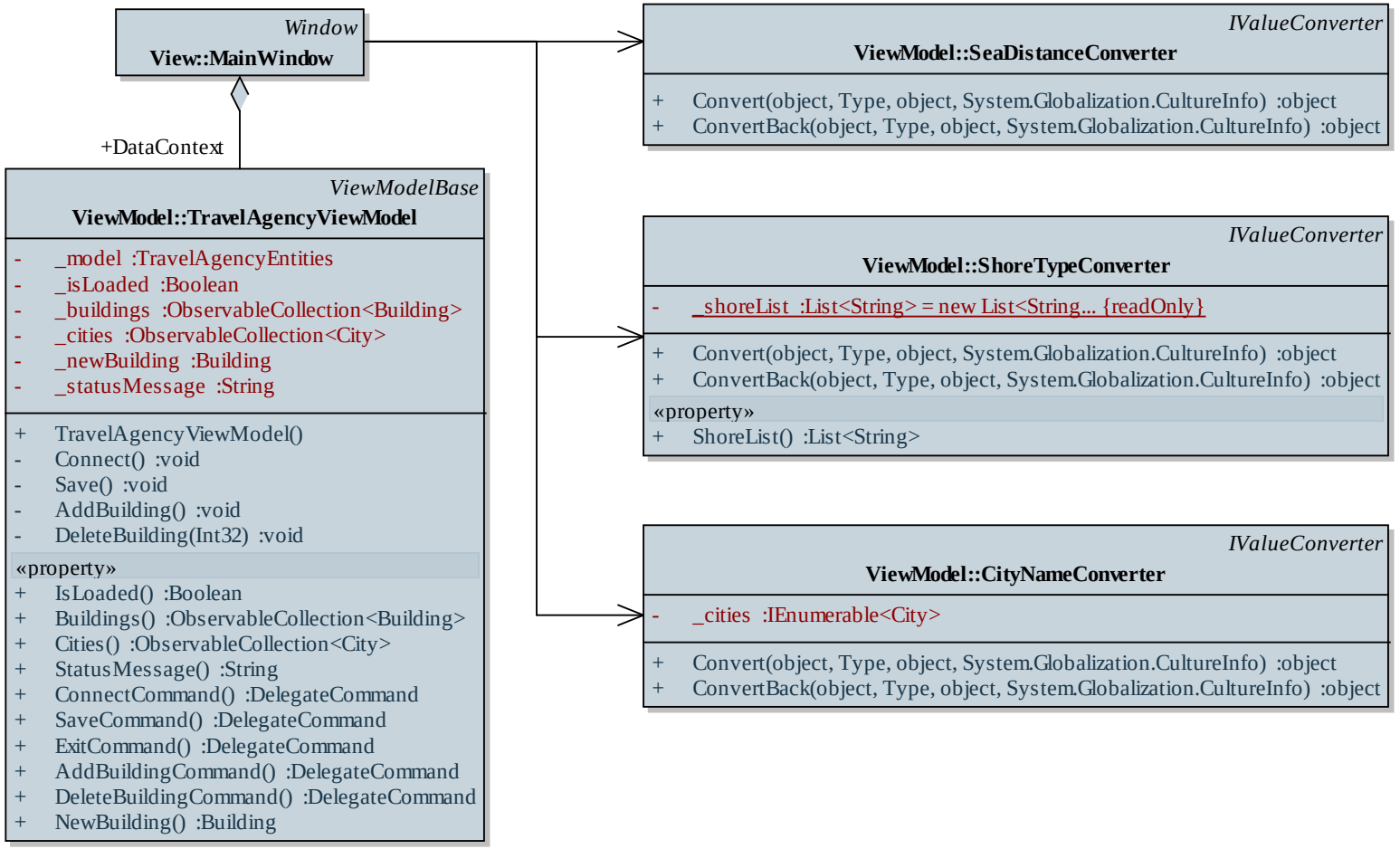
Feladat: Készítsünk el az utazási ügység épületek (**`Building`**) táblájának karbantartó alkalmazását, ahol grafikus felületen keresztül szerkeszthetjük a tartalmat.

- a rácsban csak a név, városnév, tengerpart távolság, tengerpart típus és megjegyzés értékeket jelenítjük meg, utóbbi kettőhöz speciális átalakító szükséges (**`SeaDistanceConverter`**, **`ShoreTypeConverter`**)
- új épület felvételéhez külön szerkesztőmezőket hozunk létre, ahol a városok neveit szintén átalakítással jelenítjük meg (**`CityNameConverter`**)
- a hozzáadáshoz felveszünk egy új épületet (**`NewBuilding`**), amely köthető a felülethez

Az adatkezelés eszközei

Példa

Tervezés:



Az adatkezelés eszközei

Példa

Megvalósítás (`MainWindow.xaml`*):*

```
…
<DataGrid Name="buildingGrid" Grid.Row="1" …
          ItemsSource="{Binding Buildings}">
<!-- adatrács, amelynek megadjuk az oszlopait -->
   <DataGrid.Columns>
      …
      <DataGridTextColumn
            Header="Tengerpart távolság"
            Binding="{Binding SeaDistance,
              Converter={StaticResource
                seaDistanceConverter}}" />
      <!-- konverzió használata -->
      …
```

Az adatkezelés eszközei

Példa

*Megvalósítás (***ShoreTypeConverter.cs***):*

```
…
public object Convert(object value, …) {
   if (value == null || !(value is Int32))
      // ellenőrizzük az értéket
      return Binding.DoNothing;
      // ha nem megfelelő, nem csinálunk semmit

  Int32 index = (Int32)value;
  if (index < 0 || index >= _shoreList.Count)
     return Binding.DoNothing;

  return _shoreList[index];
}
```

Az adatkezelés eszközei

Példa

Megvalósítás (`SeaDistanceConverter.cs`*):*

```
public object ConvertBack(object value, …) {
   if (value == null || !(value is String))
      // ellenőrizzük az értéket
      return DependencyProperty.UnsetValue;
         // ha nem megfelelő, nem tudjuk az
         // értéket beállítani

   if (_shoreList.Contains(value as String))
      return _shoreList.IndexOf(value as String);
   else
      return 0;
}
```

Az adatkezelés eszközei

Számított adatok

- Az entitásmodell által biztosított típusok parciális (**`partial`**) osztályok, ezért tetszőlegesen kiegészíthetőek
  - lehetőségünk van új, számított tulajdonságok hozzáadására, (felhasználhatjuk az idegen-kulccsal társított adatokat is)
  - lehetőségünk van a típus metódusokkal való felruházására
- Pl.:

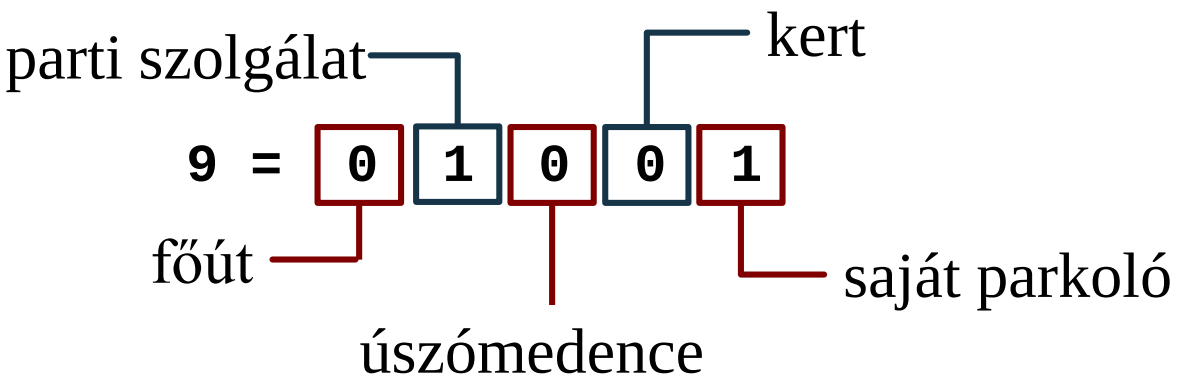
```
public partial class Customer { // kiegészítések
   public Boolean IsLivingInCountry {
      get { return Address.City != "Budapest"; }
      // felhasználjuk a csatolt adatokat
   }
}
```

Az adatkezelés eszközei

Példa

Feladat: Javítsunk az épületek megjelenítésén úgy, hogy látható legyen az apartmanok száma, valamint lehessen szerkeszteni a jellemzőket is.

- a jellemzők bináris formában tárolják az épület tulajdonságait (kert, parkoló, …) annak érdekében, hogy a későbbiekben könnyen fel tudjunk venni új tulajdonságokat a táblaszerkezet módosítása nélkül
- pl.:



Az adatkezelés eszközei

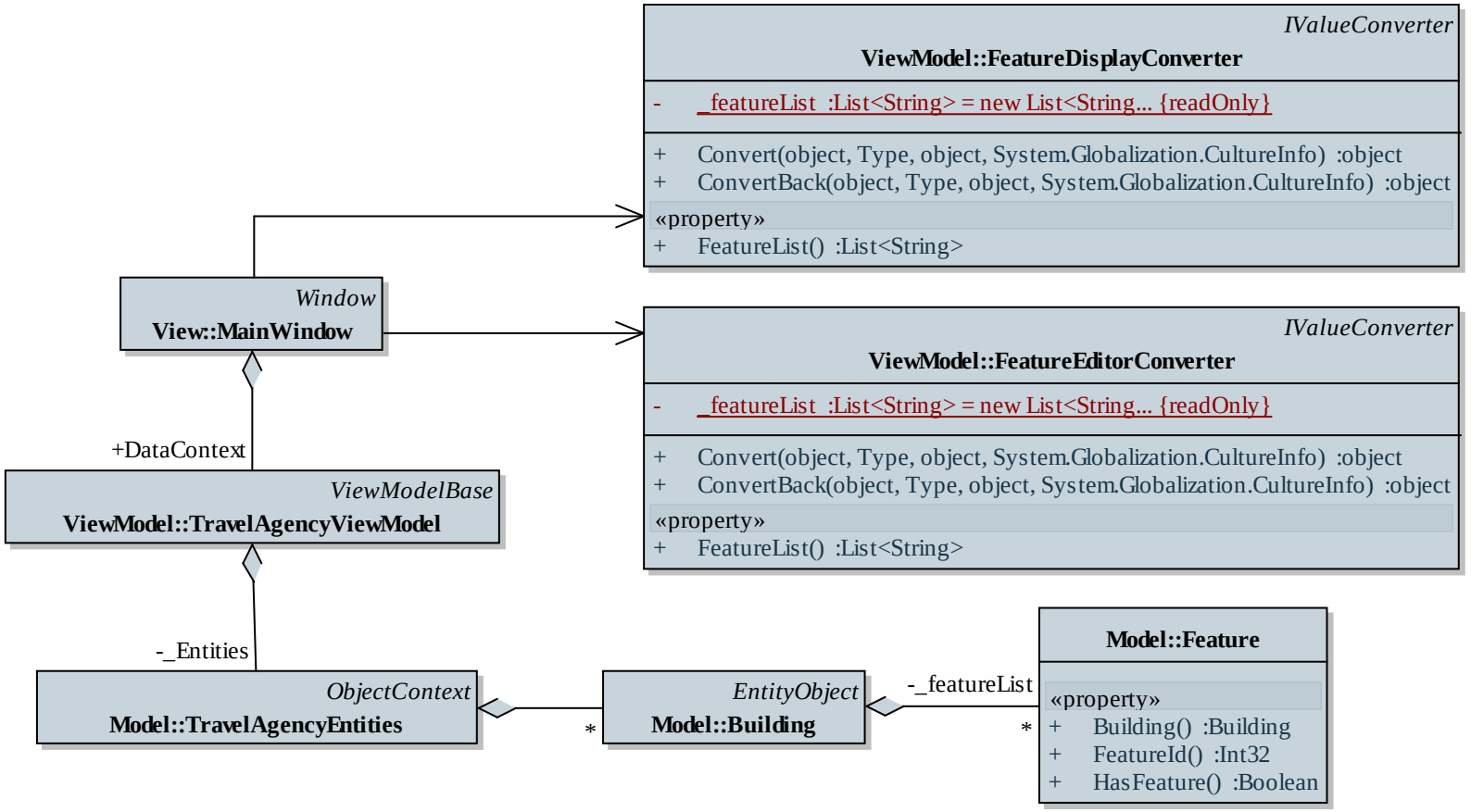
Példa

- a jellemzők megjelenítéséhez, szerkesztéséhez egy egyedi oszloptípus szükséges, amelyben a szerkesztőmező egy kijelölő mezőkből álló lista
- az apartmanok számának lekéréséhez bevezetünk egy számított oszlopot (**`ApartmentCount`**)
- hasonlóan a jellemzőket is egy külön oszlopban kell kezelnünk (**`FeaturesList`**)
  - bevezetünk egy új típust (**`Feature`**), amely megadja a jellemző sorszámát és állapotát
- a jellemző megjelenítéséhez, illetve a szerkesztéséhez külön konverzió szükséges (**`FeatureDisplayConverter`**, **`FeatureEditorConverter`**)

Az adatkezelés eszközei

Példa

Tervezés:



Az adatkezelés eszközei

Példa

Megvalósítás (`Building.Extensions.cs`*):*

```
public partial class Building {
   private List<Feature> _FeatureList;

   // apartmanok száma
   public Int32 ApartmentCount {
      get { return Apartment.Count; }
   }
   // jellemzők listája
   public List<Feature> FeaturesList {
      get { … return _FeatureList; }
   }
}
```

Az adatkezelés eszközei

Példa

Megvalósítás (`MainWindow.xaml`*):*

```
…
<DataGridTemplateColumn Header="Jellemzők">
   <DataGridTemplateColumn.CellTemplate>
      <!-- megjelenítő mező sablonja --> …
   </DataGridTemplateColumn.CellTemplate>
   <DataGridTemplateColumn.CellEditingTemplate>
      <!-- szerkesztőmező sablonja -->
      <DataTemplate>
          <!-- a szerkesztőmező egy lista lesz -->
          <ListBox ItemsSource="{Binding
                                 FeaturesList}">
          …
```

Az adatkezelés eszközei

Adatok szűrése/rendezése

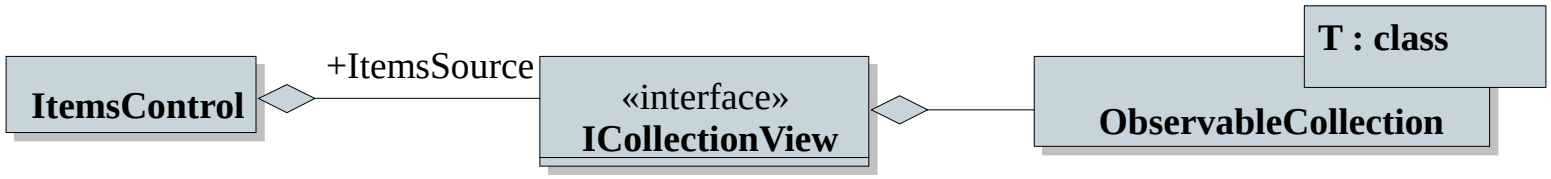
- Adatok szűrése/rendezése megvalósítható LINQ segítségével
  - eleve szűrt/rendezett tartalmat biztosítunk a felügyelt gyűjtemény számára
  - pl.:

```
ObservableCollection<Customer> customers =
   new ObservableCollection<Customer>(
      db.Customers.Where(c => c.Address.City ==
                                 "Budapest")
        .SortBy(c => c.Name)
    // szűrés végrehajtása a lakhelyre, és az
    // eredmény rendezése név szerint
);
// a gyűjtemény rendezett és szűrt lesz
```

Az adatkezelés eszközei

Gyűjteménynézetek

- Lehetőségünk van automatikus szűrést, rendezést, csoportosítást és kiválasztást alkalmazni a *gyűjteménynézet*ek (**`ICollectionView`**) segítségével
  - egy helyettes (proxy), amely megfelelően továbbítja az adatokat
  - bármely gyűjteményre létrehozható a **`CollectionViewSource.GetDefaultView(…)`** metódussal
  - több típusa is adott az egyes gyűjteményekre (pl. **`ListCollectionView`**)



Az adatkezelés eszközei

Gyűjteménynézetek

- biztosítja a szűrést (**`Filter`**)
- megadható rendezési feltételek listája (**`SortDescriptions`**)
  - tetszőleges sok feltétel adható, amely egymás után kerül alkalmazásra
- az elemek csoportosíthatóak (**`GroupDescription`**)
- lehetőséget ad a vezérlőben kijelölt elem (**`CurrentItem`**) követésére a nézetmodellben
  - kódban is módosítható (**`MoveCurrentToFirst`**, **`MoveCurrentToNext`**, …)
  - ehhez szinkronizálni kell a kijelölést a vezérlőben (**`IsSynchronizedWithCurrentItem`**)

Az adatkezelés eszközei

Gyűjteménynézetek

- Pl.:

```
ObservableCollection<Customer> customers = …;
CustomersView =
   CollectionViewSource.GetDefaultView(customers);
   // nézet létrehozása a gyűjteményre

collectionView.Filter = (x => (x as Cusotmer)
   .Address.City == "Budapest");
   // szűrés az elemekre

collectionView.SortDescriptions.Add(
   new SortDescription("Name",
      ListSortDirection.Ascending));
      // új rendezési feltételek megadása
```

Az adatkezelés eszközei

Gyűjteménynézetek

- Pl.:

```
<ListView ItemsSource ="{Binding CustomersView}"
   IsSyncronizedWithCurrentItem="True">
   <!-- listás megjelenítő a nézetben, amely
        szinkronizált az aktuális elemmel -->
…
<Button Content="Elejére" Command=… />


…
private void GotoFirst(){ // parancs végrehajtása
   CusotrmersView.MoveCurrentToFirst();
      // a nézetmodellben is szabályozható lesz a
      // felületi kijelölés
}
```

Az adatkezelés eszközei

Példa

Feladat: Valósítsuk meg az apartmanok megjelenítését (adott épület kiválasztására), valamint az épületek szűrését város alapján.

- felveszünk egy gyűjteménynézetet az épületekre, amelyet legördülő menüben történő kiválasztás alapján szűrünk
- felveszünk még egy adatrácsot, amelynek tartalma az aktuálisan kiválasztott épülethez tartozó apartmanok lesznek (adatkötéssel könnyen megadható)
- az apartmanok megfelelő megjelenítéséhez további konverziók szükségesek (**`PriceConverter`**, **`DayOfWeekConverter`**)

Az adatkezelés eszközei

Példa

Megvalósítás (`TravelAgencyViewModel.cs`*):*

```
…
Buildings = new ObservableCollection<Building>(
   _model.Building.Include("Apartment")
             .Include("City"));
BuildingsView = CollectionViewSource
             .GetDefaultView(Buildings);
   // szűréshez szükséges nézet példányosítása a
   // gyűjteményre
…
BuildingsView.Filter = (building =>
    (building as Building).City.Name.Equals(obj)
); // szűrő felvétele
…
```

Az adatkezelés eszközei

Példa

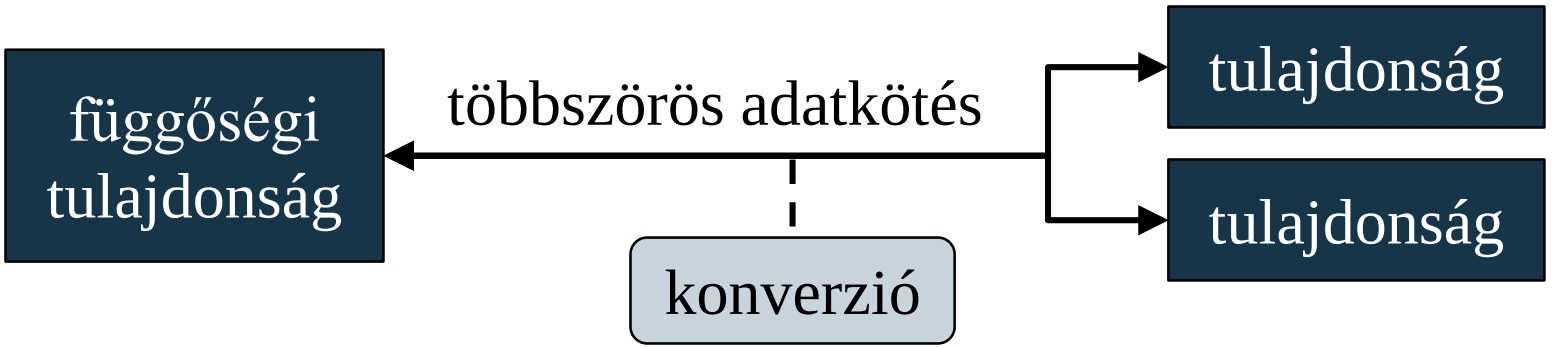
Megvalósítás (`DayOfWeekConverter.cs`*):*

```
…
public object Convert(object value, …,
                      CultureInfo culture){
   …
   return culture.DateTimeFormat
        .DayNames[(Int32)value - 1];
      // lekérjük a nap nevét az aktuális nyelvi
      // környezetben
   …
```

Az adatkezelés eszközei

Többszörös kötés és konverzió

- Lehetőségünk van egy függőségi tulajdonságra több tulajdonságot is kötni (**`MultiBinding`**)
  - több egyszerű kötés (**`Binding`**) gyűjteménye
  - csak megfelelő konverzióval (**`IMultiValueConverter`**) jeleníthetőek meg az adatok
    - tömbként fogadja (a kötés sorrendjében) az adatokat, és ugyanebben a sorrendben kell visszaadnia



Az adatkezelés eszközei

Többszörös kötés és konverzió

- Pl.:

```
<TextBlock>
   <TextBlock.Text>
      <!-- szöveg összetett megadása -->
      <MultiBinding Converter="…">
         <!-- többszörös kötés átalakítóval -->
         <Binding Path="…" />
         <Binding Path="…" />
         <!-- tetszőleges sok kötést adunk meg -->
      </MultiBinding>
   </TextBlock.Text>
</TextBlock>
```

Az adatkezelés eszközei

Többszörös kötés és konverzió

- Pl.:

```
class MyMultiConverter : IMultiValueConverter {
    public object Convert(object[] values, …){
        // egy tömbben kapjuk meg az értékeket, a
        // megadott kötések sorrendjében
        …
    }
    public object[] ConvertBack(object value, …){
        // egy tömbben szolgáltatjuk vissza az
        // eredményt, ismét a megadott sorrendben
        …
    }
}
```

Az adatkezelés eszközei

Példa

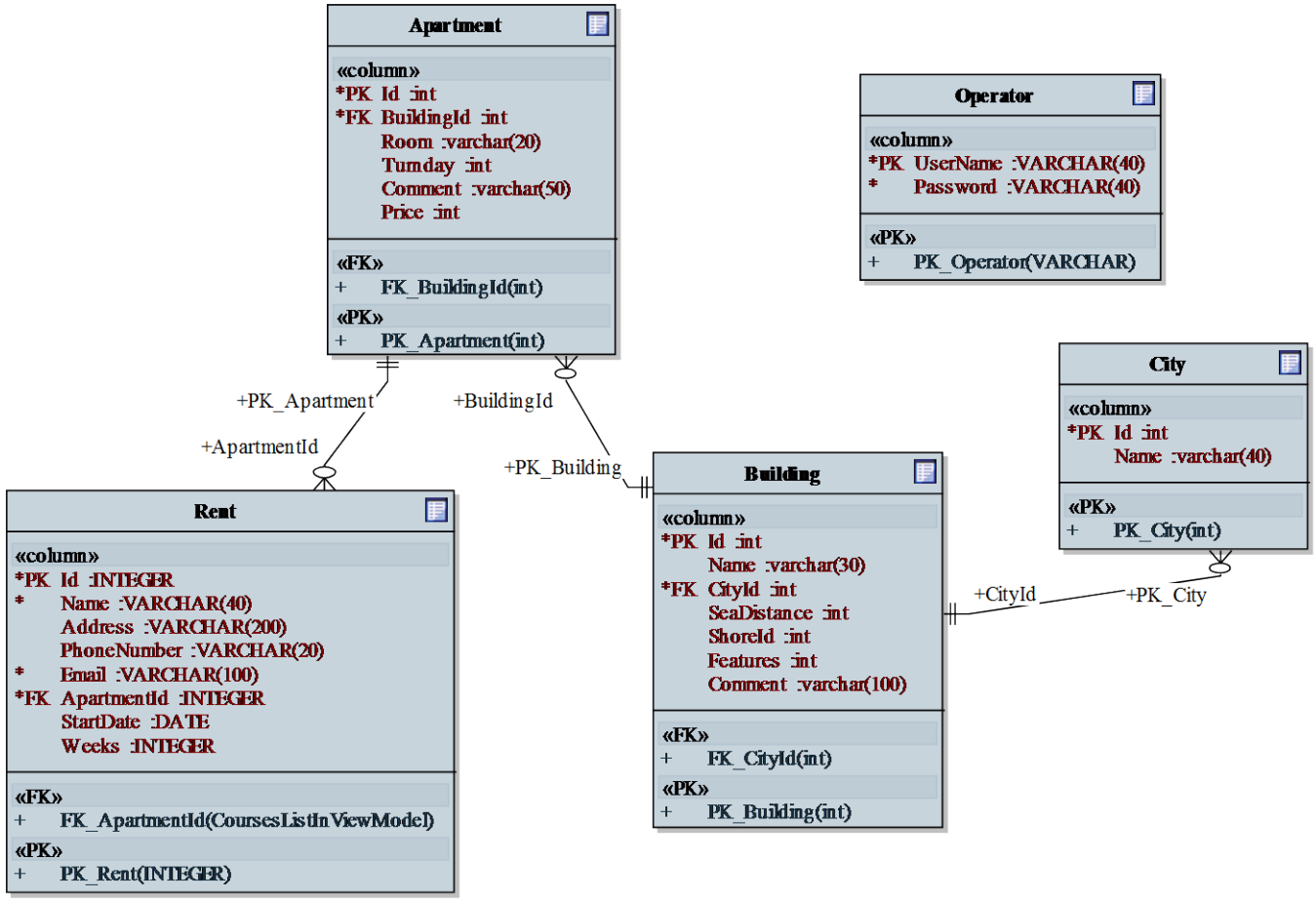
Feladat: Valósítsuk meg az apartman foglalási funkciót. Listázzuk ki az apartman kiválasztásával a hozzátartozó foglalásokat, és lehessen törölni, vagy újat létrehozni.

- a létrehozást külön nézetben végezzük az adatok megadásával (**`NewRentWindow`**)
- az adatbázisban a foglalásnál a kezdődátum és a hetek száma van eltárolva, de megjeleníteni a kezdő és a végdátumot szeretnénk, így többszörös adatkötés szükséges
- a foglalás mentésekor több ellenőrzést kell végeznünk, a foglaló e-mail címének formátumát is ellenőrizzük
- felhasználó-kezelést is bevezetünk, egy új nézetben (**`LoginWindow`**) kérjük be az adatokat

Az adatkezelés eszközei

Példa

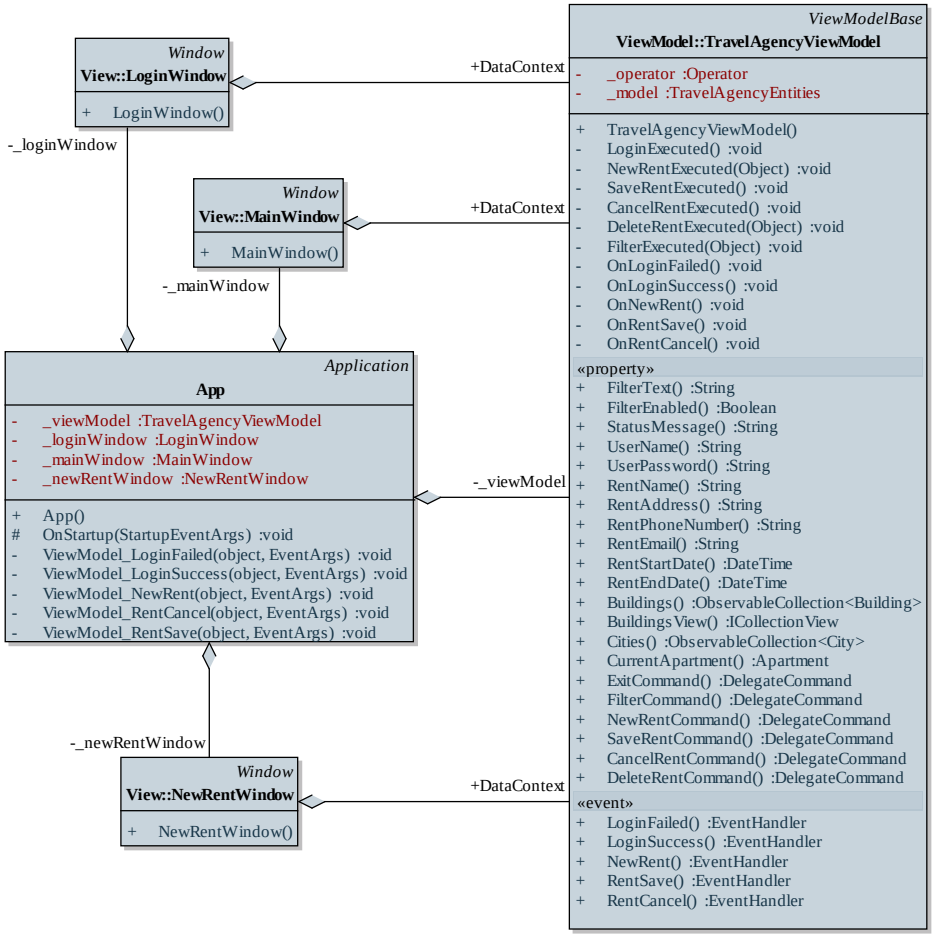
Tervezés (adatbázis):



Az adatkezelés eszközei

Példa

Tervezés (architektúra):



Az adatkezelés eszközei

Példa

Megvalósítás (`MainWindow.xaml`*):*

```
…
<DataGrid Name="rentGrid" …>
   <DataGrid.Columns> …
      <DataGridTextColumn Header="Vége dátum">
         <DataGridTextColumn.Binding>
            <MultiBinding
               Converter="{StaticResource
                           endDateConverter}">
               <Binding Path="StartDate" />
               <Binding Path="Weeks" />
            </MultiBinding>
         </DataGridTextColumn.Binding>
         …
```

Az adatkezelés eszközei

Példa

Megvalósítás (`MainWindow.xaml`*):*

```
…
_operator =
    _model.Operator.FirstOrDefault(op =>
        op.LoginName == UserName &&
        op.Password == UserPassword);
    // lekérjük a felhasználót a megadott értékek
    // segítségével
if (_operator == null) {
    // nem található megfelelő felhasználó
    OnLoginFailed();
        // jelezzük, hogy sikertelen a bejelentkezés
    return;
}
```