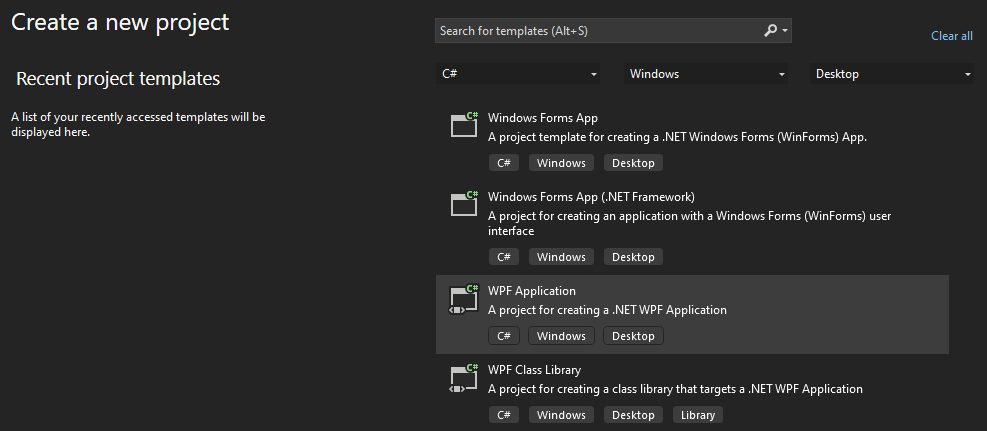
03-03-01-grid-and-simple-binding

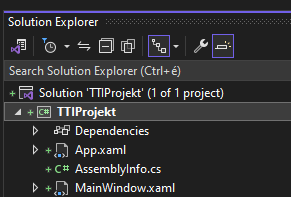
# Projekt létrehozás

1. Hozzuk létre a TTIProjekt-et a megadott tárhelyen!

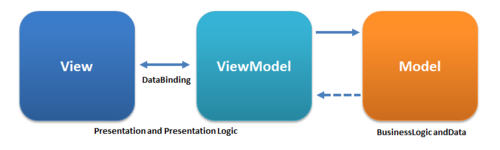


2.

Eredmény:

****

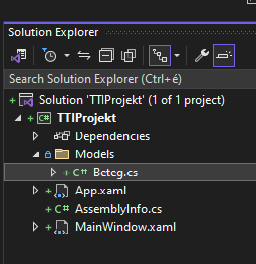
# MVVM model



## Model réteg

A model réteg jelen esetben az az adatstuktúrát (név, magasság, súly) és üzleti logikát (tti számítás) fogja tárolni.

Létrehozzuk a Models mappát és abban a Beteg osztályt.



Osztályok: <https://www.w3schools.com/cs/cs_classes.php>

# Commitok

A feladat összes [commit](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commits/03-03-01-grid-and-simple-binding)-ja.

## Commitok egyenként

Beteg.css

1. Beteg osztály: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/af9f103c88dbc6324a8936e3e6466bc951abd1dd)

Az osztály public (minenki számára látható) legyen.

**public** class Beteg

{

}

1. Beteg osztály adattagjai: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/b103da95bcd2750a614361a7270d6fc54dc70846)

Használhatjuk a névkonvenciók esetén az „\_” jelet az adattagok nevének megadásakor.

Névkonvenciók: <https://www.c-sharpcorner.com/UploadFile/8a67c0/C-Sharp-coding-standards-and-naming-conventions/>

public class Beteg

{

private string **\_**nev;

private double **\_**suly;

private double **\_**magassag;

1. Alapértelmezett konstuktor: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/753fc2079d7c42a439ca2ca14a5e5595bf84bdeb)

### View réteg – Grid (táblázatos elrendezés)

* Grid ismertetése: <https://wpf-tutorial.com/panels/grid-rows-and-columns/>
* TextBlock conrol: <https://wpf-tutorial.com/hu/15/alap-vezerlok/a-textblock-vezerlo-sorok-formazasa/>

MainWindow.xaml

1. Grid margó: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/77f83d452633e7e776bf5d4d5af4d1b947fb383a)
2. Grid felépítése: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/69cc39775a2c171048e64ebbd1eed852a8af60b2)
3. Első oszlop feliratai: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/5ce231ea57369e60ed0674532e2e1ebdceb816b2)
4. Ablak méretezése: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/c15809bc6ae91b7a62226fdf27a81db0bca9d3d2)
5. Adatbeviteli mezők: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/15c7228741b2abf8c8c41304d6016650ebf76419)
6. Csak olvasható beviteli mezők: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/f1924889c6edce9e338553cb00a49a5e6f46c805)

## Model réteg

1. Írható tulajdonságok: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/c16df0925140975a41d430d5ef04c264951ebdb1)
2. Olvasható tulajdonságok: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/ae3ae6b9e2164bda1b79e5f676cc2c17d59c94ba)

## View réteg

1. Model réteg névtéránek megadása a view-n: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/89a4743eb7688657fd4dc327cc4a409d7ea1cb75)
2. A view réteg adatkontextusa (data context) egy beteg objektum lesz: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/3ad1624e8adea8eb04dede99216b1ab13214e469)

Data context: <https://wpf-tutorial.com/hu/36/data-binding/using-the-datacontext/>

A projektet ezután build-eljük le, hogy a kapcsolat a két réteg között létrejöjjön.

1. A beteg nevének kötése

Data binding: <https://wpf-tutorial.com/hu/35/data-binding/hello-bound-world/>

UpdateSourceTrigger: <https://wpf-tutorial.com/hu/37/data-binding/the-updatesourcetrigger-property/>

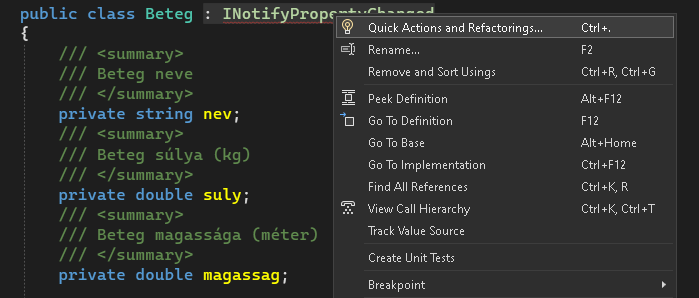
1. Beteg súlyának kötése: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/4dc7cee8b7ee75acb88a9aeb6e10e35efdd93cfb)
2. Beteg magasságának kötése: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/bdbad2348b36f996a28b2c36c2219bb0e9478559)
3. TTI és BetegAdatok kötése: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/598039532f3708b5d2e2584a01c2ae1825325940)

# Model réteg

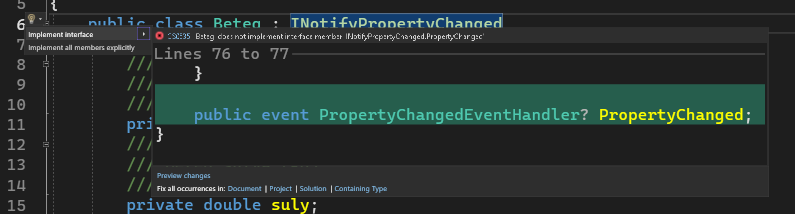
Futtassuk a programot! Sajnos az eredmények nem jelennek meg.

1. INotifyPropertyChanged interface: szeretnénk, ha a változások megtörténnének view-n is: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/33060f78dc928865382f79d199f49690b4aa3994)
2. Az interfac-t implementáljuk:

a)



b)



1. Az interfac-t implementáltuk

Az új kód a Beteg osztályban megjelent:

public event PropertyChangedEventHandler? PropertyChanged;

1. Megírjuk az interface szükséges metódusát: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/805df712f3f7fe6a9ba27cb326c7cf2f2de1ab69)

Mindig ezt kell írni!

1. Jelezzük melyik adatbevitelnél melyik gettelhető property változik: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/2e0b7fc36febfc751ed55fc759d834eb81e115f5)
2. Kötjük az eredményeket megjelenítő TextBoxokat, és csak olvashatóvá tesszük: [link](https://github.com/csarp-dotnet-core-wpf/03-03-wpf-data-binding/commit/16313ae65e1b27cb13bfc7657210941369eb037d)

Megoldandó feladatok:

1. Név, súly, magasság kezdőértékei jelenjenek meg a program indulásakor!
2. Ha a név üres, ne jelenjen meg a beteg adatok mezőben semmi!
3. A beteg adatok mezőben legyen valamilyen értékelése az eredménynek (<https://hu.wikipedia.org/wiki/Testtömegindex> )
4. Az adatbevitelkor jelenjen meg a kg és méter mértékegység