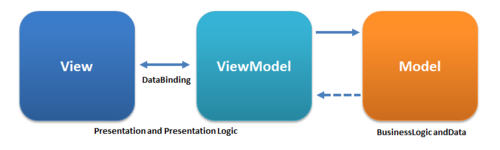
# 11- Architektury .NET

# .NET

* **.NET Framework** je originální implementace .NET
* Podporuje vytváření stránek, serviců (idk) a desktopových aplikací na Windows
* **.NET** je multiplatformní implementace .NET podporující vytváření webových stránek, serviců a konzolových aplikací na Windows, Linux a macOS (před .NET 5 .NET Core)
* **Xamarin/Mono** je .NET implementace pro spouštění aplikací na Android a iOS
* **.NET Standard** je API společné pro všechny .NET implementace (umožňuje spouštění stejného kódu a knihoven na různých implementacích)
* .NET Framework aplikace jsou napsány v C#, F#, VB nebo v jiném z .NET jazyku a jsou zkompilovány do **Common Intermediate Language (CIL)**
* **Common Language Runtime (CLR)** rozběhne .NET aplikaci na daném zařízení a zkonvertuje CIL na strojový kód (umí také spravovat paměť, kontrolovat datové typy, řeší výjimky, poskytuje garbage collector a thread-safeness)
* **Framework Class Library (FCL)** je kolekce všech tříd, interface, datových typů a knihoven obsažených v .NET (věšinou s namespacu System nebo Microsoft)

# MVVM (Model View ViewModel)

* Mimo rodinu .NET také znám jako Model View Binder
* Softwarová architektura (lze považovat za návrhový vzor)
* Oddělení GUI (View) od logiky aplikace (Model)
* ViewModel se stará o převedení dat z Modelu do podoby prezentovatelné ve View, používá logiku svou a modelu
  + Nemá odkaz na View, neví o něm (narozdíl od Presenteru v MVP), objevuje se v něm redundantní kód (boilerplate)
* View neobsahuje žádný logický kód, s ViewModelem komunikuje pomocí bindingu
  + Bežným jazykem je XAML (eXtensible Application Markup Language)

# Binding

* Vytvoření vazby mezi vlastnostmi poskytovatele (ViewModel) a konzumenta (View)
* Data jsou synchronizována
* Poskytovatel při změně upozorní všechny konzumenty (WPF/UWP – INotifyPropertyChanged)
* Existuje také obousměrný Binding: {Binding Foo, Mode TwoWay}

# Command

* Událost připravena na pozdější použití zapozdřena do objektu
* Metoda, respektive delegát, který je v commandu uchován, je volán, pokud nastane nějaká událost ve View
* WPF/UWP – ICommand(Execute, CanExecute)

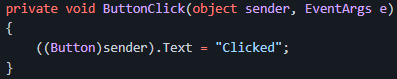
# Converter

* Metoda, která modifikuje binding data – Přetypování svázaných dat na jiný datový typ
* Většinou převod klasických typů (bool, int, ...) na typy View (barva, obrázek, ...)

# Observer

* Návrhový vzor, závislost 1 : 0..N mezi poskytovatelem a konzumenty
* Při změně poskytovatele jsou upozorněni všichni konzumenti a mohou na změnu reagovat
* Příkladem tohoto NV je právě binding
  + Žádný nebo několik prvků ve View jsou pozorovatelé jedné vlastnosti ve ViewModel
  + Změna této vlastnosti upozorní pozorovatele, ti zavolají její getter a podle něj se aktualizují

# Událostmi řízené programování (event-driven)

* Obecnější pojem označující typ asynchronního programování, je základním principem tvorby GUI aplikací
* Tok programu je řízen událostmi, které jsou obvykle spuštěny určitou uživatelskou akcí
* Základní princip – V programu běží smyčka kontrolující nastání událostí a volající patřičné metody
* sender – komponenta, od které přišla událost
  + Typ je object, je tedy potřeba přetypovat nebo dotázat se na konkrétnější typ operátorem is
* e – obsahuje data o události