Module 1
Design Windows Desktop aplikací

MGR. TOMÁŠ HAVETTA - MCT

Obsah

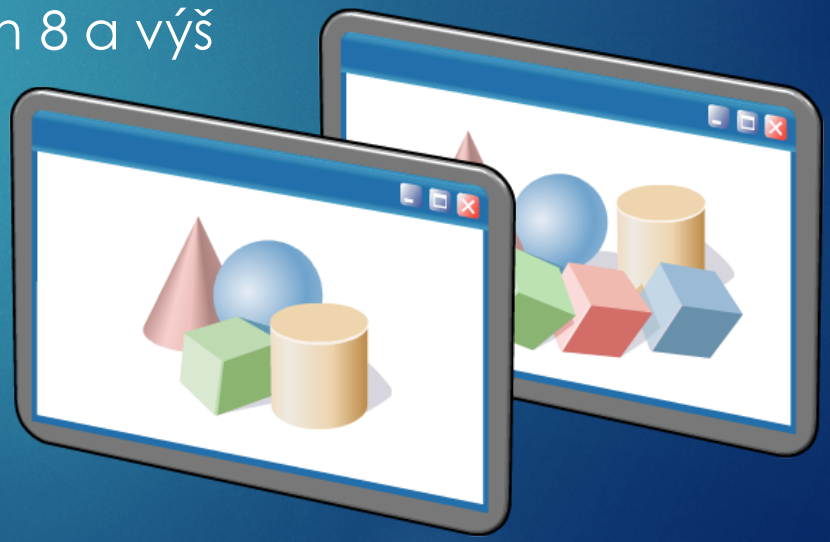
- Windows Desktop technologie
- Architectural Patterns

Blok 1: Windows Desktop technologie

- Historie vývoje Windows desktop aplikací
- Windows Forms
- Windows Presentation Foundation
- Universal Windows Platform
- Srovnání Windows Desktop technologií
- Výhled do budoucna

Historie vývoje Windows desktop aplikací

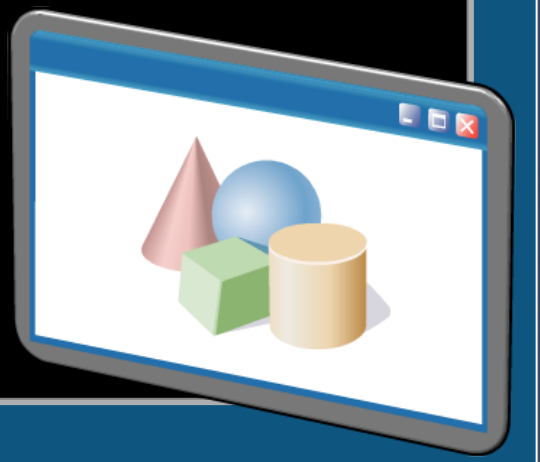
- Win32 – od 1995 (C/C++)
- MFC (C++) a VB, systémy třetích stran (Delphi, Symantec)
- Windows Forms – od 2002 .NET Framework 1.0
- WPF – od 2007 .NET Framework 3.0
- UWP – od 2012, vyžaduje Win 8 a výš
- WF a WPF na .NET 5 (2020)



Windows Forms

Klíčové prvky a vlastnosti:

- User interface controls
- Settings
- Událostmi řízené UI
- Podporuje Windows Resources
- Podpora pro System dialog boxes
- Možnost kreslení GDI+
- Data binding
- Multithreading a použití background vláken
- Deployment (MSI, ClickOnce)



Umístění tříd pro Windows Forms:

- **System.Drawing**
- **System.Windows.Forms**

Windows Presentation Foundation

Klíčové prvky a vlastnosti

- XAML-based UI
- Data binding
- Podpora 2-D a 3-D grafiky
- Multimédia a animace
- Podpora pro tvorbu sestav a tisk
- Řízení UI pomocí commands a routed events
- Interoperabilita s Windows Forms controls

Architektura WPF:

- PresentationFramework
- PresentationCore
- milcore

Universal Windows Platform

Klíčové prvky a vlastnosti

- UI => XAML, WinUI, HTML, DirectX
- Data binding
- Podpora různých zařízení (Win10, Xbox, Tablety)
- WinRT
- Vhodné pro on-line aplikace
- Adaptabilita UI dle zařízení
- Podpora MS Store pro distribuci

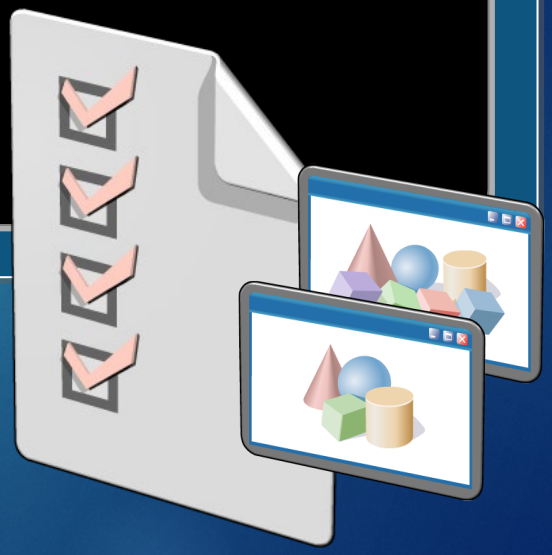
Architektura WPF:

- PresentationFramework
- PresentationCore
- milcore

Srovnání Windows Desktop technologií

Vaše možnosti:

- Windows Forms na .NET Framework 4.x
- Windows Presentation Foundation na .NET Framework 4.x
- WF/WPF na .NET Core 3.1 nebo .NET 5
- Universal Windows Platform
- Kombinace Windows Forms a WPF



Výhled do budoucna

- WinUI 3 – UWP
- Project Reunion (preview) – nezávislé na OS

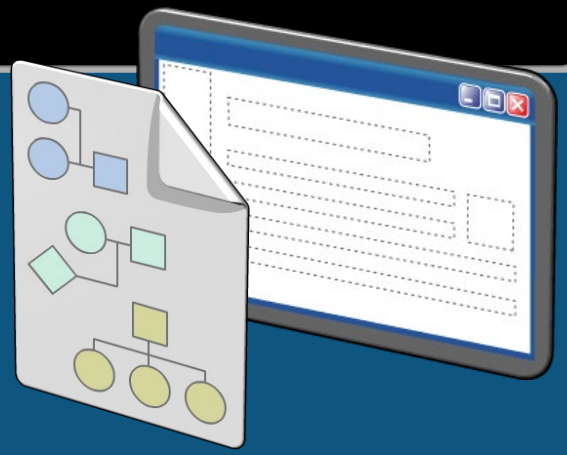
Blok 2: Architectural Patterns

- Proč Design Patterns
- UI Design Patterns

Proč Design Patterns?

Design patterns:

- Jsou formalizované postupy řešení problému
- Můžou abstraktně popsat časté problémy
- Mohou abstraktní formou popsat řešení
- Popisují kdy daný pattern použít

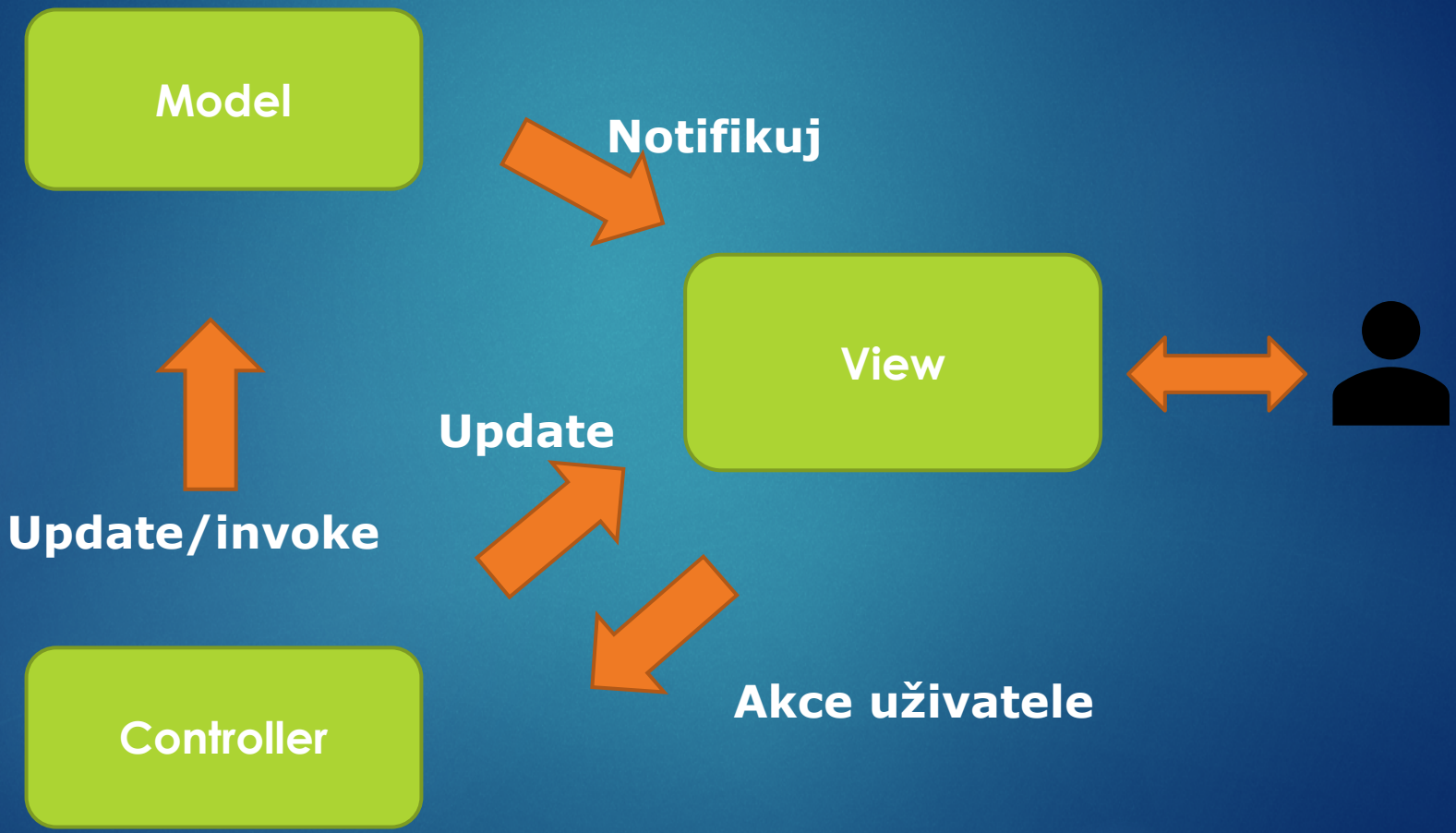


UI Design Patterns

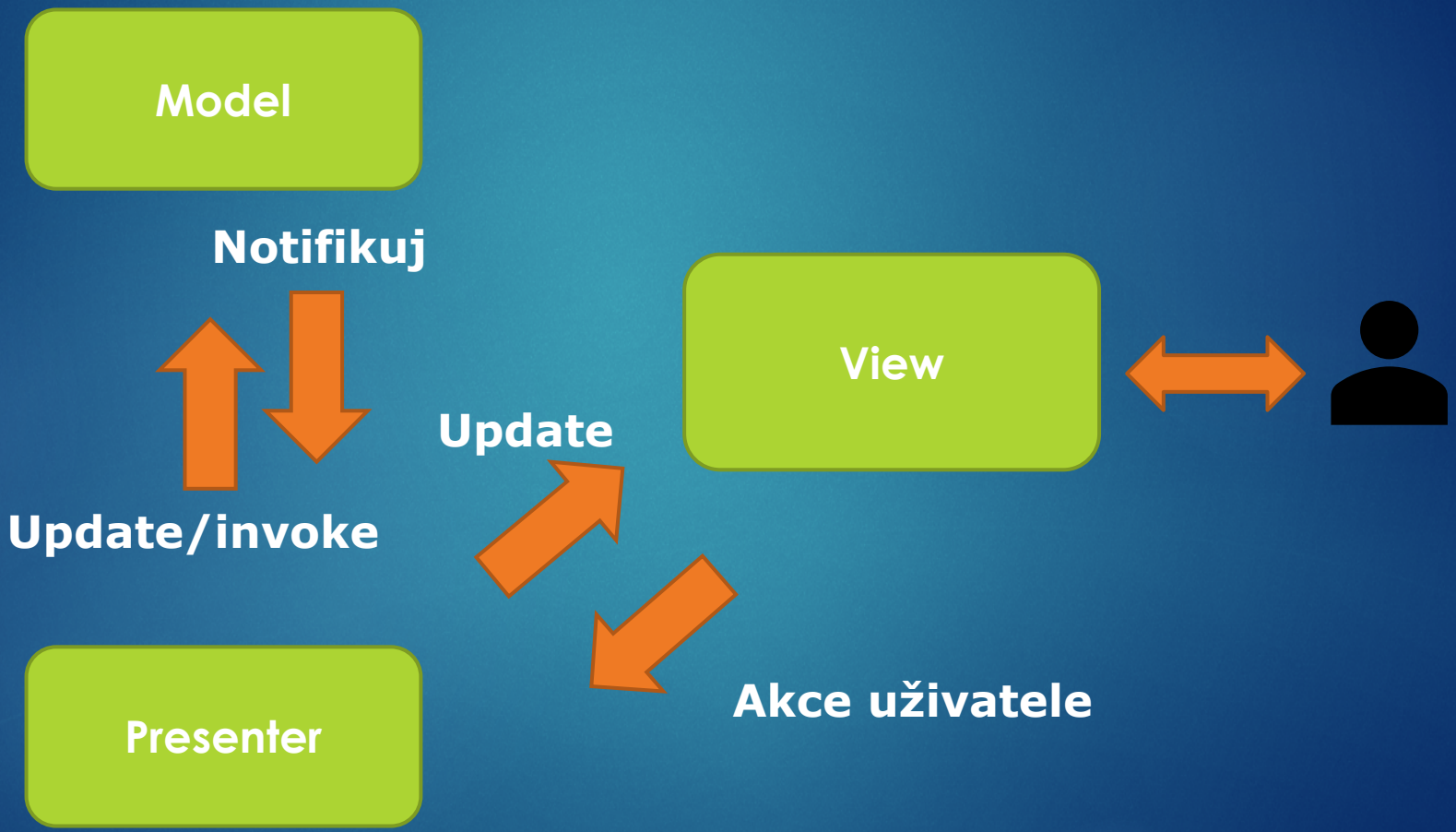
Presentation patterns:

- Model-View-Controller
  - Decouples UI od data modelu
  - Často používané u Web aplikací
- Model-View-Presenter
  - Vyvynulo se z MVC pro event-driven aplikace
  - Vhodné pro forms-over-data vývoj (WF, mobil)
- Model-View-ViewModel
  - Navrženo na základě MVP patternu
  - Vhodné pro WPF applications

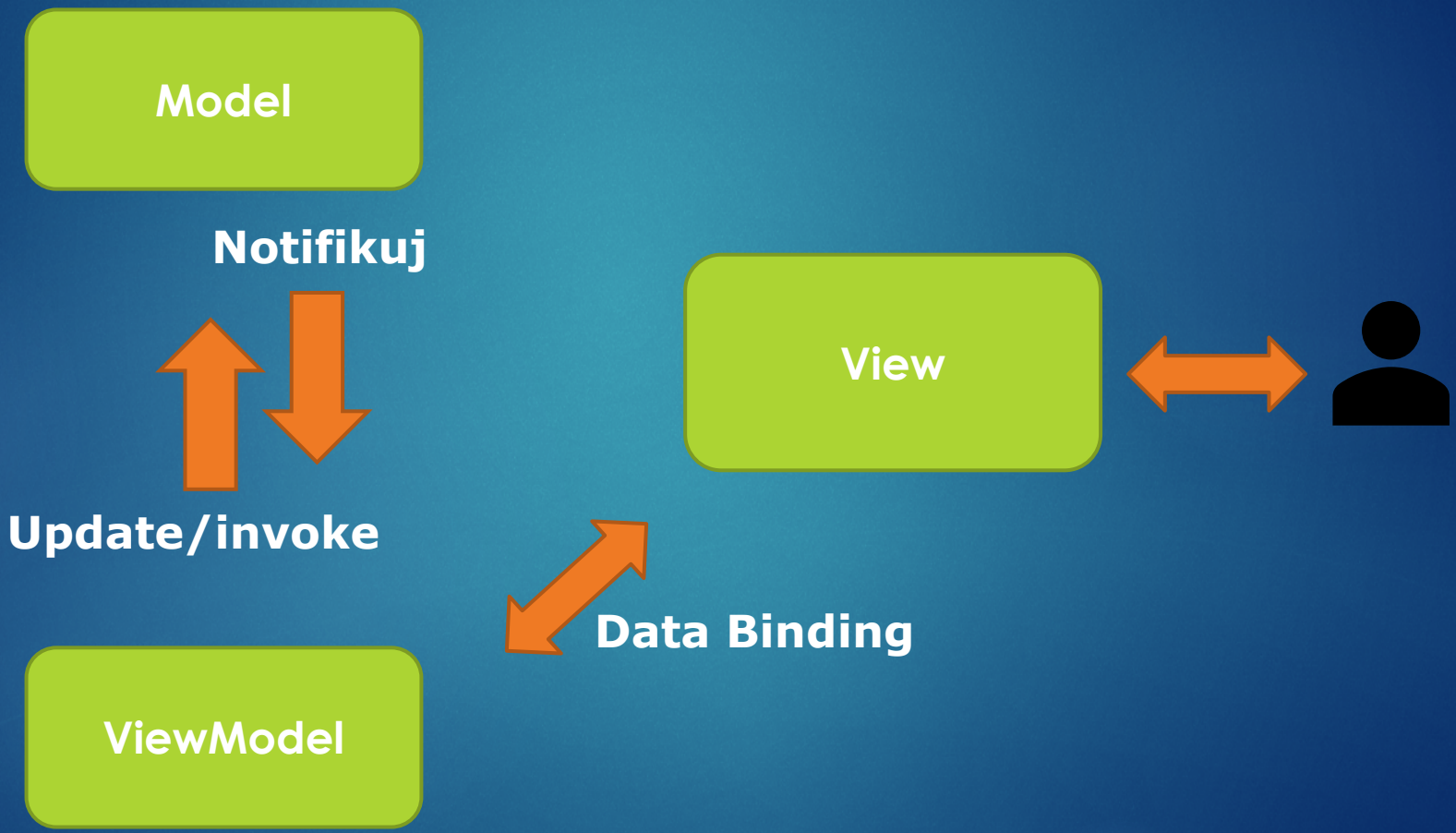
MVC


Model
Notifikuj
View
Update
Update/invoke
Controller
Akce uživatele

MVP – passive view



MVVM


Model
Notifikuj
Update/invoke
View
Data Binding
ViewModel

Lab: Volba Windows desktop technologie

- Sepište si jaká kritéria použijete při volbě technologie pro vývoj aplikace a proč je dané kritérium relevantní a jakou má váhu. Seřaďte je dle váhy.
- Máte stávající Windows desktop LoB aplikaci, která je napsaná pomocí WinForms, pro přístup k databázi používá ADO.NET. Aplikace má minimum grafických informací, téměř vše je textem.
  - Team A => sestavte důvody, proč to přepsat do WPF nebo UWP
  - Team B => najděte důvody, proč dál pokračovat ve vývoji pomocí Windows Forms