Module 4
Testování, Unit Testy a Debugg

MGR. TOMÁŠ HAVETTA - MCT

Obsah

- Testovací strategie
- Debug XAML
- Neošetřené výjimky

Blok 1: Testovací strategie

- Jak testovat Windows aplikaci
- Možné strategie testování
- Unit testy a Visual Studio
- UI Automatizované testy

Jak testovat Windows aplikaci

- Proč testovat
  - Snížení nákladů
  - Ujištění se, že se aplikace chová dle očekávání
  - Snížení nákladů na provoz
- Typy testů
  - Unit testy !!!
  - Integrační testy
  - Regresní testy
  - Akceptační testy
  - …
- TDD

Možné strategie testování

- Unit testy
  - Framework pro UT
- Design Patterns které zlepšují možnosti Unit testů
  - MVVM
  - IoC – Inversion of Control
  - DI – Dependency Injection
  - Striktní používání interface
- UI Automation
  - Simuluje interaktivní testy (klikání, psaní)
  - Náročné udržovat hotové testy

Unit testy a Visual Studio

- VS 2019 podporuje přímo frameworky na unit testy
  - MSTest
  - NUnit
  - xUnit
  - …
- Dle použitého frameworku se vytváří knihovny testů. Pro nastavení testů slouží atributy

UI Automatizované testy

- Umožňují simulovat činnost uživatele v UI

```
// Vytvoření Automation instance z Button1.
AutomationElement button =
AutomationElement.RootElement.FindFirst(
    TreeScope.Descendants,
    new PropertyCondition(
        AutomationElement.AutomationIdProperty,
        "Button1"));

// Vytvoření odpovídající činnosti pattern objektu.
InvokePattern pattern = (InvokePattern)
    button.GetCurrentPattern(InvokePattern.Pattern);

// Simulace kliknutí na tlačítko.
pattern.Invoke();
```

Blok 2: Debug XAML

- Debug pomocí VS
- Demo
- WPF a Tracing

Debug pomocí VS

- Klasické možnosti debuggeru
- WPF Tree Visualizer
- Tracing pomocí **PresentationTraceSources**
- Pouze v .NET 5
  - Hot Reload – možnost změny XAML v debugované aplikaci
  - Bindings Error – dohledání chyb v Bindingu

Demo

WPF a Tracing

- **PresentationTraceSources.TraceLevel** lze nastavit pro třídy odvozené z:
  - **BindingBase**
  - **BindingExpressionBase**
  - **DataSourceProvider**
- Hodnoty pro **TraceLevel**
  - None
  - Low
  - Medium
  - High

Blok 3: Neošetřené výjimky

- WPF a výjimky
- Jak zalogovat neošetřené výjimky

WPF a výjimky

- Část chyb (např. Binding) WPF ignoruje
- Neošetřené výjimky skončí pádem aplikace
- Všechna místa, kde se dá očekávat výjimka by měla být ošetřena try blokem
- Jednotný mechanizmus informace o výjimkách
- **Logování – KAM a CO!**

Jak zalogovat neošetřené výjimky

- Ošetření nezpracovaných WPF výjimek

```
this.Dispatcher.UnhandledException +=
    new DispatcherUnhandledExceptionEventHandler(
        OnDispatcherUnhandledException);
```

- Ošetření jakýchkoliv nezpracovaných výjimek

```
AppDomain.CurrentDomain.UnhandledException +=
    new UnhandledExceptionEventHandler(
        OnUnhandledException);
```

Lab: Debug a výjimky

- Otestujte debug okna aplikace Prevodnik a projděte si Visual Tree View. Zkontrolujte u prvků, že některé vlastnosti jsou nastaveny stylem.
- Zpracujte neošetřenou výjimku vzniklou zadáním nesmyslu do textboxu