Konwertery i walidacja w wiązaniu

Czas: 2 × 45 min | praca indywidualna | Wykład 2.04

Cel: Zbudujesz formularz rejestracji, który waliduje dane w modelu przez INotifyDataErrorInfo, pokazuje komunikaty pod polami i używa konwerterów do koloru i widoczności.

Po zajęciach potrafisz:

- implementuje INotifyDataErrorInfo w modelu
- definiuje szablon błędu Validation.ErrorTemplate w stylu
- pisze własny IValueConverter i używa BooleanToVisibilityConverter
- blokuje zapis przy błędach danych

Sprzęt i materiały

- komputer z Windows 11 i dostępem do internetu
- pendrive lub konto w repozytorium do zapisania projektu

Oprogramowanie

- **Visual Studio Community 2026 (obciążenie „Programowanie aplikacji klasycznych platformy .NET")** — darmowe dla uczniów i nauczycieli; w pracowni zainstalowane visualstudio.microsoft.com/pl/downloads
- **.NET 10 SDK (LTS)** — instalowany razem z Visual Studio; sprawdzenie: `dotnet --version` dotnet.microsoft.com/pl-pl/download/dotnet/10.0

Bezpieczeństwo

- Zapisuj projekt w swoim folderze (np. `C:\Users\<login>\source\repos`), a na koniec zajęć skopiuj go na pendrive lub wyślij do repozytorium.
- Nie uruchamiaj programów i pakietów z nieznanych źródeł; pakiety NuGet instaluj tylko te, które wskazuje zadanie.
- Rób przerwy dla oczu (zasada 20-20-20) — praca z projektantem okien wymaga długiego patrzenia w ekran.

Przebieg zadania

1 Model z walidacją

Utwórz projekt WPF `Rejestracja` i klasę `Formularz`.

1. Właściwości `Imie`, `Email`, `Wiek` (string — własny komunikat zamiast błędu konwersji).
2. Reguły jako metody statyczne zwracające tablicę błędów: imię min. 2 litery, e-mail wyrażeniem regularnym, wiek liczba 13–120.
3. Implementacja `HasErrors`, `GetErrors`, `ErrorsChanged` (lekcja 2.04).
4. `MoznaZapisac` — true, gdy pola wypełnione i brak błędów; `Niepelnoletni` — wiek < 18.

2 Szablon błędu

Komunikat pod polem zamiast samej czerwonej ramki.

```
<ControlTemplate x:Key="BladPodPolem">
  <StackPanel>
    <AdornedElementPlaceholder />
    <TextBlock Foreground="#B42318" FontSize="12"
               Text="{Binding [0].ErrorContent}" />
  </StackPanel>
</ControlTemplate>
```

1. W zasobach okna zdefiniuj `ControlTemplate` z `AdornedElementPlaceholder` i TextBlock `{Binding [0].ErrorContent}`.
2. Styl TextBox: dolny margines na komunikat i `Validation.ErrorTemplate`.
3. Sprawdź komunikaty dla: „O”, „ola@”, „dwanaście”, „121”.

3 Konwertery

Wygląd zależny od danych.

1. Konwerter `WiekNaKolorConverter`: wiek < 18 — pomarańczowy, ≥ 18 — zielony, brak — szary; użyj go dla koloru tekstu pola wieku.
2. Komunikat „potrzebna zgoda rodzica” widoczny tylko dla niepełnoletnich (BooleanToVisibilityConverter).
3. Przycisk Zapisz z `IsEnabled="{Binding MoznaZapisac}"`.

4 Sprawdzenie

Testy ręczne.

1. Wypełnij tabelę testów (min. 8 przypadków).
2. Zrób zrzuty: błędy pod polami, niepełnoletni, poprawny formularz.
3. Zatwierdź pracę w Git.

W sprawozdaniu umieść

- ☐ kod klasy Formularz i konwertera
- ☐ XAML z szablonem błędu
- ☐ tabela testów
- ☐ zrzuty

Pytania kontrolne

1. Dlaczego wiek przechowujemy jako string?
2. Co zwraca GetErrors i kiedy WPF ją wywołuje?
3. Czym różni się konwerter od StringFormat?
4. Dlaczego BooleanToVisibilityConverter nie pozwala odwrócić warunku i jak to obejść?

Kryteria oceny

Kryterium	Punkty
INotifyDataErrorInfo z regułami	6
Szablon błędu w stylu	4
Konwerter koloru i widoczność	4
MoznaZapisac i testy	3
Pytania	3
Razem	**20**

Sprawozdanie przygotuj na wzorze dołączonym do zadania (plik .docx o tym samym numerze).