Wiązanie danych — przelicznik jednostek

Czas: 2 × 45 min | praca indywidualna | Wykład 2.01

Cel: Zbudujesz przelicznik odległości i temperatury, w którym kontrolki nie mają ani jednej metody obsługi zdarzeń — wszystko łączy wiązanie danych z modelem implementującym INotifyPropertyChanged.

Po zajęciach potrafisz:

- implementuje INotifyPropertyChanged w klasie bazowej z metodą SetProperty
- wiąże TextBox, Slider i TextBlock z modelem przez DataContext
- dobiera tryb wiązania i UpdateSourceTrigger
- wyszukuje błędy wiązań w oknie XAML Binding Failures

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

Utwórz projekt WPF `Przelicznik` i dwie klasy bez WPF.

```
public double Kilometry
{
    get => _kilometry;
    set
    {
        if (value < 0) value = 0;
        if (SetProperty(ref _kilometry, value))
        {
            OnPropertyChanged(nameof(Mile));
            OnPropertyChanged(nameof(Metry));
        }
    }
}
```

1. `ObservableBase` z metodami `OnPropertyChanged` i `SetProperty` (lekcja 2.01).
2. `PrzelicznikModel`: `Kilometry` (wartości ujemne zamień na 0), `Mile` z getterem i setterem (1 mila = 1,609344 km), `Metry` tylko do odczytu.
3. `Celsjusz`, `Fahrenheit` (do odczytu i zapisu) i `Kelwin`.
4. Przy zmianie właściwości bazowej zgłaszaj zmiany właściwości zależnych (`OnPropertyChanged(nameof(Mile))`).

2 Okno

Zbuduj okno bez metod obsługi zdarzeń.

1. W konstruktorze okna: `DataContext = new PrzelicznikModel();`.
2. Slider 0–100 i TextBox kilometrów związane z `Kilometry`; TextBox mil związany z `Mile`.
3. TextBlock z metrami: `StringFormat=To {0:N0} metrów`.
4. Część temperatury: dwa pola tekstowe i TextBlock z kelwinami.
5. ProgressBar związany z suwakiem przez `ElementName`.

3 Tryby i błędy

Eksperymenty z wiązaniem.

1. Porównaj `UpdateSourceTrigger=PropertyChanged` i domyślny `LostFocus` w polu mil — zapisz różnicę.
2. Ustaw `Mode=OneWay` w polu kilometrów i sprawdź, co się dzieje przy pisaniu.
3. Wpisz tekst `abc` w pole kilometrów — co pokazuje okno i co trafia do modelu?
4. Wprowadź literówkę w ścieżce (`Kilometr`) i znajdź komunikat w oknie **XAML Binding Failures** (Debugowanie → Okna). Popraw.
5. Zatwierdź pracę w Git.

W sprawozdaniu umieść

- ☐ kod modelu
- ☐ kod XAML okna (bez zdarzeń)
- ☐ tabela wyników eksperymentów
- ☐ zrzut komunikatu XAML Binding Failures

Pytania kontrolne

1. Dlaczego właściwość Mile wymaga zgłoszenia zmiany przy zmianie Kilometry?

2. Czym różni się UpdateSourceTrigger=LostFocus od PropertyChanged?
3. Co się stanie z wiązaniem, gdy model nie implementuje INotifyPropertyChanged?
4. Kiedy użyjesz wiązania ElementName?

Kryteria oceny

Kryterium	Punkty
ObservableBase i model z zależnościami	6
Okno wyłącznie na wiązaniach	6
Eksperymenty z trybami i błędami	5
Pytania	3
Razem	**20**

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