Kalkulator WPF

Czas: 2 × 45 min | praca indywidualna | Wykład 1.26

Cel: Zbudujesz kalkulator z przyciskami cyfr i działań, w którym jedna metoda obsługuje wszystkie przyciski, a stan kalkulatora przechowuje osobna klasa.

Po zajęciach potrafisz:

- obsługuje wiele przycisków jedną metodą (parametr `sender`, `Tag`, zdarzenia trasowane)
- przechowuje stan aplikacji w klasie logiki
- prowadzi historię działań w ListBox
- obsługuje klawiaturę zdarzeniem KeyDown (rozszerzenie)

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 Logika kalkulatora

Utwórz projekt WPF `Kalkulator` i klasę `KalkulatorLogika` (bez WPF).

1. Właściwość `Ekran` (string, początkowo „0").
2. Metody: `Cyfra(char)`, `Przecinek()`, `Dzialanie(char)`, `Rownasie()` (zwraca wpis do historii), `Wyczysc()`.
3. Dzielenie przez zero — komunikat na ekranie zamiast wyjątku; następny klawisz czyści błąd.
4. Działania wykonuj od lewej do prawej, jak w prostym kalkulatorze (2 + 3 × 4 = 20).

2 Okno

Ekran (TextBox tylko do odczytu, tekst do prawej) i siatka przycisków.

1. Użyj `UniformGrid` 5 × 4 z przyciskami C, ÷, ×, −, cyfry, +, =, przecinek.
2. Każdy przycisk ma `Tag` ze znakiem dla logiki (np. Content="×" Tag="*").
3. Po prawej: ListBox historii i przycisk czyszczenia historii.

3 Jedna metoda dla wszystkich przycisków

Zamiast 17 metod — jedna.

```
private void Przycisk_Click(object sender, RoutedEventArgs e)
{
    if (e.OriginalSource is Button b && b.Tag is string znak)
        Obsluz(znak[0]);
}
```

1. Podłącz zdarzenie na kontenerze: `<UniformGrid ButtonBase.Click="Przycisk_Click">`.
2. W metodzie odczytaj przycisk z `e.OriginalSource` i znak z `Tag`, a następnie wywołaj odpowiednią metodę logiki.
3. Po każdym klawiszu przepisz `Ekran` do TextBoxa; wynik `=` dopisz na górze historii.
4. Przetestuj: 12 + 7,5 =; 10 ÷ 4 =; 1 ÷ 0 =; 0,1 + 0,2 =; 7 ÷ 3 =.

4 Klawiatura (rozszerzenie)

Obsługa klawiatury numerycznej.

1. Dodaj zdarzenie `KeyDown` okna i zamień klawisze (`Key.NumPad0`… `Key.Add`, `Key.Enter`, `Key.Escape`) na znaki.
2. Ustaw `e.Handled = true` po obsłużeniu klawisza.
3. Zatwierdź pracę w Git.

W sprawozdaniu umieść

- ☐ kod klasy KalkulatorLogika
- ☐ kod obsługi przycisków
- ☐ tabela testów
- ☐ zrzut kalkulatora z historią

Pytania kontrolne

1. Czym różni się `sender` od `e.OriginalSource` w zdarzeniu trasowanym?
2. Do czego służy właściwość `Tag`?
3. Dlaczego 0,1 + 0,2 daje w tym kalkulatorze dokładnie 0,3?
4. Jak zmienić logikę, aby mnożenie miało pierwszeństwo przed dodawaniem?

Kryteria oceny

Kryterium	Punkty
Klasa logiki z obsługą błędów	6
Okno z UniformGrid i Tag	4
Jedna metoda obsługi przycisków	4
Historia i testy	3
Klawiatura lub pytania	3
Razem	**20**

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