Kontrolki wyboru — konfigurator pizzy

Czas: 2 × 45 min | praca indywidualna | Wykład 1.25

Cel: Zbudujesz konfigurator pizzy z przyciskami opcji, polami wyboru, listą rozwijaną i suwakiem, a cenę obliczy osobna klasa logiki.

Po zajęciach potrafisz:

- odczytuje stan kontrolek RadioButton, CheckBox, ComboBox i Slider
- aktualizuje wynik przy każdej zmianie wyboru
- oddziela obliczenia od interfejsu (klasa Cennik)
- zabezpiecza obsługę zdarzeń zgłaszanych podczas wczytywania okna

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 Klasa Cennik

Utwórz projekt WPF `Pizzeria` i plik `Cennik.cs` (bez WPF).

1. `enum Rozmiar { Mala, Srednia, Duza }`; ceny bazowe 24 / 32 / 42 zł.
2. Metoda `Cena(Rozmiar r, int liczbaDodatkow, bool grubeCiasto, int ilosc)`: dodatek 4 zł, grube ciasto +3 zł; od 3 sztuk rabat 10% (zaokrąglenie do grosza).
3. Dla ilości spoza 1–10 zgłoś `ArgumentOutOfRangeException`.
4. Policz ręcznie: średnia, 2 dodatki, grube, 1 szt. = 43 zł; średnia, 1 dodatek, 3 szt. = 97,20 zł.

2 Okno

Zbuduj interfejs na Gridzie z dwiema kolumnami.

1. GroupBox „Rozmiar” z trzema RadioButton (wspólny `GroupName`, mała zaznaczona).
2. GroupBox „Dodatki” z WrapPanel `PnlDodatki` i co najmniej 5 CheckBox.
3. ComboBox `CmbCiasto` (cienkie, grube), Slider `SldIlosc` 1–10 z przeskokiem co 1 i wyświetlaną wartością.
4. ListBox zamówień, duży TextBlock z ceną i przycisk Zamów.

3 Cena na żywo

Cena zmienia się przy każdej zmianie wyboru.

```
private void Przelicz()
{
    // zdarzenia podczas wczytywania okna – kontrolki mogą jeszcze nie istnieć
    if (TxtCena is null || PnlDodatki is null || SldIlosc is null) return;
    TxtCena.Text = $"Do zapłaty: {AktualnaCena():C}";
}
```

1. Podłącz zdarzenia `Checked` / `Unchecked`, `SelectionChanged`, `ValueChanged` do metody `Przelicz()`.
2. Liczbę zaznaczonych dodatków policz: `PnlDodatki.Children.OfType<CheckBox>().Count(c => c.IsChecked == true)`.
3. Uruchom program. Jeśli pojawi się `NullReferenceException` przy starcie — dodaj w `Przelicz` sprawdzenie, czy kontrolki już istnieją (szablon).
4. Przycisk Zamów dopisuje do listy opis zamówienia z ceną.

4 Sprawdzenie

Porównaj ceny z obliczeniami ręcznymi.

1. Przetestuj co najmniej 5 konfiguracji i wpisz je do tabeli.
2. Zrób zrzut okna z kilkoma zamówieniami. Zatwierdź pracę w Git.

W sprawozdaniu umieść

- ☐ kod klasy Cennik
- ☐ kod XAML i obsługi zdarzeń
- ☐ tabela 5 konfiguracji z ceną ręczną i z programu
- ☐ zrzut okna

Pytania kontrolne

1. Dlaczego warunek zapisujemy `IsChecked == true`?
2. Dlaczego zdarzenie Checked może wystąpić, zanim istnieją inne kontrolki?
3. Co daje wydzielenie obliczeń do klasy Cennik?
4. Jak odczytać tekst wybranego elementu ComboBox?

Kryteria oceny

Kryterium	Punkty
Klasa Cennik z rabatem i walidacją	5
Okno z wszystkimi kontrolkami	5
Cena aktualizowana na żywo	4
Lista zamówień	2
Tabela testów i pytania	4
Razem	**20**

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