Delegaty i zdarzenia — czujnik temperatury

Czas: 2 × 45 min | praca indywidualna | Wykład 1.18

Cel: Zbudujesz klasę czujnika, która powiadamia inne części programu o nowych pomiarach i przekroczeniu progu za pomocą zdarzeń, oraz napiszesz filtr przyjmujący warunek jako parametr `Func<T,bool>`.

Po zajęciach potrafisz:

- deklaruje zdarzenie `event EventHandler<T>` i własną klasę argumentów
- subskrybuje zdarzenie wieloma metodami obsługi (`+=`)
- przekazuje funkcję jako argument (`Func<double,bool>`, wyrażenie lambda)
- wyjaśnia, po co oddzielać źródło zdarzenia od reakcji

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.

Przebieg zadania

1 Klasa argumentów i czujnik

Utwórz projekt `Czujnik`.

```
public event EventHandler<OdczytEventArgs>? PrzekroczonoProg;

public void Zmierz(double temperatura)
{
    _pomiary.Add(temperatura);
    var e = new OdczytEventArgs(temperatura);
    NowyOdczyt?.Invoke(this, e);
    if (temperatura > ProgAlarmu) PrzekroczonoProg?.Invoke(this, e);
}
```

1. Klasa `OdczytEventArgs : EventArgs` z właściwościami `Temperatura` i `Czas`.
2. Klasa `Czujnik` z nazwą, progiem alarmu i listą pomiarów (tylko do odczytu na zewnątrz).
3. Zdarzenia: `NowyOdczyt` i `PrzekroczonoProg` typu `EventHandler<OdczytEventArgs>?`.
4. Metoda `Zmierz(double t)` zapisuje pomiar, zgłasza `NowyOdczyt`, a gdy t > próg — także `PrzekroczonoProg`.

2 Subskrybenci

W programie głównym podłącz trzy reakcje.

1. Wypisywanie każdego odczytu z godziną.
2. Alarm na czerwono z nazwą czujnika (z parametru `sender`).
3. Dziennik alarmów: dopisywanie wpisu „godzina;temperatura” do listy `dziennik`.
4. Symuluj 12 pomiarów: `new Random(7)` i `22 + los.NextDouble() * 8`. Dlaczego ziarno 7 daje za każdym razem te same wyniki?
5. Odłącz alarm (`-=`) po trzecim alarmie i sprawdź, że dziennik nadal działa (potrzebna nazwana metoda zamiast lambdy).

3 Filtrowanie funkcją

Napisz metodę `Filtr(IEnumerable<double> dane, Func<double, bool> warunek)`.

1. Metoda zwraca listę elementów spełniających warunek — bez LINQ, pętlą `foreach`.
2. Wywołaj ją z lambdami: temperatury powyżej 27 °C i poniżej 24 °C.
3. Dodaj trzeci warunek zapisany jako zwykła metoda `static bool Normalna(double t)` i przekaż jej nazwę.
4. Porównaj wynik z `Where` z LINQ.
5. Zatwierdź pracę w Git.

W sprawozdaniu umieść

- ☐ kod klas i programu
- ☐ zrzut ekranu symulacji z alarmami
- ☐ zawartość dziennika
- ☐ wyniki filtrów

Pytania kontrolne

1. Co oznacza zapis `NowyOdczyt?.Invoke(this, e)` i dlaczego jest `?.`?
2. Czym różni się delegat `Func<double,bool>` od `Action<double>`?
3. Dlaczego lambdy nie da się łatwo odłączyć operatorem `-=`?
4. Podaj przykład zdarzenia z aplikacji okienkowej.

Kryteria oceny

Kryterium	Punkty
Klasa argumentów, czujnik i zdarzenia	6
Trzech subskrybentów i odłączanie	5
Metoda Filtr z Func i trzy warunki	5
Pytania	4
Razem	**20**

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