Grafika i animacja — zegar analogowy

Czas: 2 × 45 min | praca indywidualna | Wykład 2.09

Cel: Narysujesz zegar analogowy na Canvas z gradientem i transformacjami obrotu, napędzany DispatcherTimer, oraz dodasz animację Storyboard.

Po zajęciach potrafisz:

- rysuje kształty i podziałkę na Canvas
- obraca elementy RotateTransform związanym z ViewModelem
- aktualizuje interfejs cyklicznie DispatcherTimer
- tworzy animację DoubleAnimation wyzwalaną zdarzeniem

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
- **Pakiet NuGet CommunityToolkit.Mvvm 8.4** — instalacja: Zarządzaj pakietami NuGet nuget.org/packages/CommunityToolkit.Mvvm

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 Obliczenia

Projekt WPF `Zegar`.

1. Klasa statyczna `KatyZegara` z metodami `Sekundy`, `Minuty`, `Godziny` (DateTime → kąt w stopniach).
2. Sprawdź ręcznie i w krótkim teście: 12:30:00 → 15°, 180°, 0°; 21:45:15 → 292,5°, 271,5°, 90°.
3. `ZegarViewModel` z właściwościami kątów i czasu cyfrowego oraz metodą `Ustaw(DateTime)`.

2 Tarcza

Canvas 300 × 300.

```
<Line X1="150" Y1="150" X2="150" Y2="40" Stroke="#26243A" StrokeThickness="4">
  <Line.RenderTransform>
    <RotateTransform Angle="{Binding KatMinut}" CenterX="150" CenterY="150" />
  </Line.RenderTransform>
</Line>
```

1. Koło z `RadialGradientBrush` i obramowaniem.
2. 12 kresek podziałki (dłuższe co 3 godziny) — współrzędne oblicz: x = 150 + r·sin(α), y = 150 − r·cos(α).
3. Trzy wskazówki jako `Line` od środka w górę z `RotateTransform Angle="{Binding KatMinut}" CenterX="150" CenterY="150"`.
4. Środek zegara — małe koło.

3 Ruch i animacja

Zegar ożywa.

1. DispatcherTimer co 250 ms wywołuje `Ustaw(DateTime.Now)`.
2. Czas cyfrowy pod zegarem (czcionka Consolas).
3. Przycisk Start/stop z animacją powiększenia po najechaniu (EventTrigger MouseEnter, AutoReverse).
4. Zatwierdź pracę w Git.

W sprawozdaniu umieść

- ☐ obliczenia kątów dla 3 godzin
- ☐ kod XAML tarczy
- ☐ zrzut zegara
- ☐ wyjaśnienie, dlaczego DispatcherTimer, a nie System.Timers.Timer

Pytania kontrolne

1. Jak obliczyć kąt wskazówki godzinowej?
2. Dlaczego obrót wymaga podania CenterX i CenterY?
3. Co się stanie, gdy zmienisz właściwość kontrolki w zdarzeniu timera z innego wątku?
4. Do czego służy AutoReverse?

Kryteria oceny

Kryterium	Punkty
Obliczenia kątów i ViewModel	5
Tarcza z gradientem i podziałką	5
Wskazówki z RotateTransform	4
Timer i animacja	4
Pytania	2
Razem	**20**

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