Klient REST API — kursy walut NBP

Czas: 2 × 45 min | praca indywidualna | Wykład 2.16

Cel: Zbudujesz aplikację pobierającą aktualną tabelę kursów walut z API Narodowego Banku Polskiego, z przelicznikiem, historią notowań i odporną na brak sieci.

Po zajęciach potrafisz:

- pobiera dane JSON asynchronicznie przez HttpClient
- mapuje odpowiedź na rekordy DTO
- obsługuje błędy sieci, limitu czasu i formatu
- oddziela logikę przeliczeń od komunikacji sieciowej

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
- **API NBP (bez rejestracji)** — kursy średnie: tabela A api.nbp.pl

Przygotowanie przed zajęciami

- Przykładowe odpowiedzi API (`pliki/tabela_A_przyklad.json`, `pliki/EUR_last5_przyklad.json`) — do pracy bez sieci i testów.

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 Rozpoznanie API

Otwórz w przeglądarce adres tabeli A.

1. `https://api.nbp.pl/api/exchangerates/tables/A/?format=json` — zapisz nazwy pól i ich typy.
2. `https://api.nbp.pl/api/exchangerates/rates/A/EUR/last/10/?format=json` — historia jednej waluty.
3. Rekordy DTO: `Kurs`, `TabelaKursow`, `KursDnia`, `KursyWaluty` (data jako DateOnly, kurs jako decimal).

2 Usługa i logika

Projekt WPF `Kursy`.

```
public async Task<TabelaKursow> PobierzTabeleAAsync(CancellationToken token = default)
{
    var tabele = await Http.GetFromJsonAsync<List<TabelaKursow>>(
        "exchangerates/tables/A/?format=json", token);
    return tabele is { Count: > 0 } ? tabele[0]
        : throw new InvalidOperationException("Pusta odpowiedź serwera.");
}
```

1. Interfejs `INbpService` i `NbpService` z jednym statycznym HttpClient (BaseAddress, Timeout 10 s).
2. Klasa statyczna `Przelicznik`: NaZlote, ZeZlotych, ZmianaProcent (zaokrąglenie do 2 miejsc).
3. Sprawdź przelicznik na danych z pliku przykładowego (100 EUR po kursie 4,4032 = 440,32 zł).

3 ViewModel i okno

Interfejs.

1. Komenda `WczytajAsync` z obsługą `HttpRequestException`, `TaskCanceledException`, `JsonException` — zrozumiałe komunikaty.
2. DataGrid kursów; przy wyborze waluty pobierz 10 ostatnich notowań i pokaż zmianę procentową.
3. Przelicznik: kwota → wynik w złotych i w walucie (właściwości odświeżane NotifyPropertyChangedFor).
4. Wskaźnik ładowania: ProgressBar widoczny podczas `WczytajCommand.IsRunning`.

4 Odporność

Sprawdź złe warunki.

1. Wyłącz sieć (tryb samolotowy lub odłączony kabel) — komunikat, brak awarii.
2. Zapytaj o nieistniejący kod waluty (np. XYZ) — co zwraca serwer (kod HTTP 404) i jak reaguje aplikacja?
3. Zatwierdź pracę w Git.

W sprawozdaniu umieść

- ☐ opis struktury JSON
- ☐ kod DTO, usługi i przelicznika
- ☐ zrzut aplikacji z kursami i historią
- ☐ wyniki testów odporności

Pytania kontrolne

1. Co to jest DTO?
2. Dlaczego HttpClient tworzymy raz?
3. Jakie wyjątki może zgłosić GetFromJsonAsync i co oznaczają?

4. Dlaczego przelicznik da się przetestować bez internetu?

Kryteria oceny

Kryterium	Punkty
DTO i usługa z HttpClient	5
Przelicznik	3
ViewModel i okno z historią	6
Obsługa błędów i testy odporności	4
Pytania	2
Razem	**20**

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