

Fetch i publiczne API — kursy walut NBP

Czas: 2 × 45 min

praca indywidualna

Cel: Napisanie aplikacji, która pobiera aktualne kursy walut z publicznego API NBP funkcją `fetch` z `async/await`, pokazuje stan ładowania i błędy, wyświetla dane w tabeli i na wykresie.

Po zajęciach potrafisz:

- pobiera dane JSON funkcją `fetch` i obsługuje je przez `async/await`
- sprawdza `response.ok` i obsługuje błędy w `try/catch/finally`
- pokazuje użytkownikowi stan „Ładowanie...” i czytelny komunikat błędu
- tworzy tabelę z filtrowaniem i wykres liniowy z danych z API
- analizuje żądania w zakładce Sieć narzędzi F12

Sprzęt i materiały

- komputer z Windows 11 i dostępem do internetu
- folder `r2_04_start` z Pliki_do_zadan (`index.html`, `styl.css`, folder `dane`)

Oprogramowanie

- **Visual Studio Code** — rozszerzenie Live Server (podgląd przez `http://127.0.0.1:5500`)
code.visualstudio.com
- **Przeglądarka Chrome, Edge lub Firefox** — narzędzia deweloperskie F12 (Elementy, Konsola, Sieć, Aplikacja)
- **API NBP** — dokumentacja; dane bez klucza i bez rejestracji
api.nbp.pl
- **Chart.js 4** — dołączany z CDN:
<https://cdn.jsdelivr.net/npm/chart.js@4.4.1/dist/chart.umd.min.js>
chartjs.org/docs/latest

Przygotowanie przed zajęciami

- Przeczytaj wykład 2.03 (Promise, `async/await`, `fetch`, CORS).

Bezpieczeństwo

- Nie wysyłaj do API zapytań w pętli ani co sekundę — pobieraj dane po załadowaniu strony i po kliknięciu „Odśwież”.
- Dane z API wstawiaj przez `textContent`; nie ufaj, że zewnętrzny serwer zawsze zwróci poprawny JSON.

Przebieg zadania

1 Poznaj API

Skopiuj `r2_04_start` do folderu `nazwisko_imie_z04` i uruchom Live Server. Adresy, których użyjesz:

```
https://api.nbp.pl/api/exchangerates/tables/A/?format=json
https://api.nbp.pl/api/exchangerates/rates/A/EUR/last/10/?format=json
```

1. Otwórz oba adresy w przeglądarce. Zapisz w sprawozdaniu: czy odpowiedź jest tablicą czy obiektem, gdzie jest data tabeli i jak nazywa się pole z kursem średnim.
2. Otwórz adres z kodem nieistniejącej waluty, np. `XYZ`, i zapisz, co zwraca serwer (F12 → Sieć: kod odpowiedzi).
3. Porównaj odpowiedź z plikami `dane/tabela_a.json` i `dane/kurs_EUR.json` — mają ten sam format (to dane przykładowe do pracy bez internetu).

2 Pobieranie tabeli kursów

Utwórz `app.js` i podłącz go na końcu `body`.

```
async function pobierzJSON(url) {
  const odp = await fetch(url);
  // jeśli !odp.ok – rzuć błąd z kodem odp.status
  return odp.json();
}
```

1. Dokończ funkcję `pobierzJSON(url)`. Napisz funkcję `adres(rodzaj, kod)`, która zwraca adres API albo — gdy zaznaczono „tryb offline” — ścieżkę do pliku w folderze `dane`.
2. Napisz `async function wczytajTabele()`: pokaż w `#stan` napis „Ładowanie...” (klasa `ładowanie`) i zablokuj przycisk „Odśwież”; pobierz tabelę A (destrukuryzacja `const [tabela] = await ...`), wpisz numer i datę tabeli do `#data-tabeli`; w `catch` pokaż komunikat z klasą `bład`; w `finally` odblokuj przycisk.
3. Wyświetl kursy w tabeli: waluta, kod, kurs z 4 miejscami po przecinku (przecinek dziesiętny). Dodaj filtrowanie na żywo polem `#szukaj` (nazwa lub kod, bez względu na wielkość liter).
4. Obsłuż przycisk „Odśwież” i pole „tryb offline” (zmiana stanu — ponowne wczytanie).

3 Wykres i przelicznik

Lista `#waluta` zawiera kody z tabeli. Po wybraniu waluty pobierz jej 10 ostatnich notowań.

1. Wypełnij listę `#waluta` (opcje „EUR — euro”), domyślnie wybierz EUR.
2. Dołącz Chart.js z CDN w `head` i napisz `wczytajWykres()`: pobierz notowania, utwórz wykres liniowy (etykiety — daty, wartości — `mid`). Przed narysowaniem nowego wykresu usuń stary metodą `destroy()`.
3. Gdy waluty nie ma w trybie offline albo serwer zwróci błąd, pokaż w `#stan-wykres` komunikat, np. „Brak notowań GBP: serwer odpowiedział kodem 404”.
4. Dodaj przelicznik: kwota w zł podzielona przez kurs wybranej waluty, wynik z 2 miejscami po przecinku i kodem waluty.

4 Testy i analiza w F12

1. W F12 → Sieć znajdź żądanie tabeli A: zapisz kod odpowiedzi, typ treści (Content-Type), rozmiar i czas. Sprawdź nagłówki odpowiedzi `Access-Control-Allow-Origin`.
2. Ustaw w zakładce Sieć ograniczenie „Slow 4G” (lub „Wolne 3G”) i zrób zrzut stanu „Ładowanie...”. Następnie wybierz „Offline” i zrób zrzut komunikatu błędu.
3. Zrób zrzut działającej aplikacji z wykresem EUR i przelicznikiem. Spakuj folder.

W sprawozdaniu umieść

- ☐ opis formatu odpowiedzi API (tablica/obiekt, nazwy pól) i odpowiedź dla waluty XYZ
- ☐ dane żądania z zakładki Sieć (kod, Content-Type, rozmiar, czas, nagłówki CORS)
- ☐ zrzuty: stan ładowania, komunikat błędu, aplikacja z wykresem
- ☐ kod funkcji `pobierzJSON` i `wczytajTabele`
- ☐ spakowany folder `nazwisko_imie_z04`

Pytania kontrolne

1. Dlaczego `fetch` nie zgłasza wyjątku przy odpowiedzi 404 i co trzeba sprawdzić samemu?
2. Do czego służy blok `finally` w funkcji `wczytajTabele`?
3. Czym jest CORS i który nagłówek odpowiedzi API NBP pozwala stronie z innej domeny odczytać dane?
4. Co by się stało, gdyby nie wywołać `destroy()` przed utworzeniem nowego wykresu na tym samym elemencie canvas?

Kryteria oceny

Kryterium	Punkty
Analiza API: format odpowiedzi, kod dla XYZ, dane z zakładki Sieć	3
<code>pobierzJSON</code> z kontrolą <code>response.ok</code> , <code>async/await</code> , <code>try/catch/finally</code> , stan „Ładowanie...”	5
Tabela kursów z formatowaniem i filtrowaniem, data i numer tabeli, tryb offline	4
Wykres <code>Chart.js</code> dla wybranej waluty z obsługą błędów i <code>destroy()</code>	4
Przelicznik walut	1
Zrzuty (ładowanie, błąd, wykres) i odpowiedzi na pytania	3
Razem	20

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