

REST API CLIENT

SPIS TREŚCI

Spis treści	1
Cel zajęć.....	1
Rozpoczęcie.....	1
Uwaga	1
Wymagania.....	2
Badanie API	2
Implementacja	2
Commit projektu do GIT.....	6
Podsumowanie.....	7

CEL ZAJĘĆ

Celem głównym zajęć jest zdobycie następujących umiejętności:

- pobieranie danych z zewnętrznych zasobów za pomocą REST API
- zdobywanie wiedzy na temat zewnętrznych API za pomocą dokumentacji typu Swagger
- wysyłanie asynchronicznych żądań z wykorzystaniem XMLHttpRequest i Fetch API

W praktycznym wymiarze uczestnicy stworzą dynamiczną stronę HTML pozwalającą na wyświetlanie bieżącej informacji pogodowej oraz prognoz dla zadanej przez użytkownika miejscowości.

ROZPOCZĘCIE

Rozpoczęcie zajęć. Powtórzenie wykonywania połączeń synchronicznych i asynchronicznych z poziomu JS na stornie.

Wejściówka? Nah

UWAGA

Ten dokument aktywnie wykorzystuje niestandardowe właściwości. Podobnie jak w LAB A wejdź do Plik -> Informacje -> Właściwości -> Właściwości zaawansowane -> Niestandardowe i zaktualizuj pola. Następnie uruchom ten dokument ponownie lub Ctrl+A -> F9.

WYMAGANIA

W ramach LAB D przygotowane powinny zostać:

- pojedyncza strona HTML ze skryptem ładowanym z zewnętrznego pliku JS
- pole tekstowe (input typu „text”) do wprowadzania adresu
- przycisk „Pogoda”, po kliknięciu którego wykonywane jest zapytanie asynchroniczne:
 - do API Current Weather: <https://openweathermap.org/current> za pomocą XMLHttpRequest
 - do API 5 day forecast: <https://openweathermap.org/forecast5> za pomocą Fetch API
- obsługa zwrotki z obu API – wypisanie pogody bieżącej oraz prognoz poniżej pola wyszukiwania.

Wygeneruj klucz do API. Ponieważ aktywacja może chwilę potrwać, na czas trwania laboratorium możesz wykorzystać „służbowy” klucz: 7ded80d91f2b280ec979100cc8bbba94. **UWAGA!** Klucz zostanie dezaktywowany niedługo po zajęciach. Musisz wygenerować swój własny.

W przypadku blokady twórczej można posiłkować się filmem: <https://www.youtube.com/watch?v=WoKp2qDFxKk> jednakże spróbuj rozwiązać ten problem samodzielnie!

Prowadzący omówi powyższe wymagania. Upewnij się, czy wszystko rozumiesz.

Tu umieść swoje notatki:

```
Json.weather  
Json.weather[0].main  
<img src = „https:....>  
...notatki...
```

BADANIE API

Poświęć kilka minut na wykonanie przykładowych zapytań do API z poziomu paska adresu przeglądarki. Podaj wymagane parametry dla osiągnięcia różnych wyników. Zbadaj odpowiedzi API, aby uzyskać pełen obraz wymagań i możliwości API.

IMPLEMENTACJA

Tradycyjnie implementację należy zacząć od zbudowania w HTML + CSS wszystkich wymaganych elementów / placeholderów na te elementy. Następnie krok po kroku należy implementować poszczególne zachowania.

Wstaw zrzut ekranu zawierającego stronę ze wszystkimi elementami, tj. pole tekstowe, przycisk, miejsce do wyświetlenia pogody i prognozy:

Pogodynka

wpisz miejscowościę

sprawdź pogodę

Bieżąca Pogoda:

pogoda na dziś....

Prognoza pogody na 5dni:

prognoza na 5dni...

Punkty:

0

1

Wstaw zrzut ekranu kodu odpowiedzialnego za wysyłanie żądania do current za pomocą XMLHttpRequest:

AI1 LAB D – Nazwisko Imię – Wersja 1

```
function getCurrentWeather(location) : void { Show usages ▾ Judyta *
  const apiKey : string = '71c0d0e44ebf4e273a3cc2213c575a67';
  const url : string = `https://api.openweathermap.org/data/2.5/weather?q=${location}&appid=${apiKey}&units=metric&lang=pl`;

  const xhr : XMLHttpRequest = new XMLHttpRequest();
  xhr.open( method: "GET", url, async: true);
  xhr.onreadystatechange = function () : void {
    if (xhr.readyState === 4 && xhr.status === 200) {
      const data = JSON.parse(xhr.responseText);
      console.log("Bieżąca pogoda:", data);
      drawWeather(data);
    }
  };
  xhr.send();
}

function drawWeather(data) : void { Show usages new *
```

Wstaw zrzut ekranu pokazujący otrzymaną odpowiedź za pomocą `console.log()` w przeglądarce.

The screenshot shows a browser's developer tools with the 'Console' tab selected. It displays the output of a `console.log(data)` statement. The output is a detailed JSON object representing the current weather in Szczecin, Poland. Key fields include `coord`, `weather` (an array containing one element), `base` (set to 'stations'), `main` (containing temperature information), `sys` (containing system details like ID and country code), and `wind` (containing wind speed and direction). The JSON is formatted with collapsible sections indicated by arrows and expanded sections indicated by dots.

```
Bieżąca pogoda: js.js:20
▼ {coord: {...}, weather: Array(1), base: 'stations', main: {...}, visibility: 10000, ...} ⓘ
  base: "stations"
  ▶ clouds: {all: 11}
  cod: 200
  ▶ coord: {lon: 14.553, lat: 53.4289}
  dt: 1731665037
  id: 3083829
  ▶ main: {temp: 6.39, feels_like: 3.8, temp_min: 5.05, temp_max: 7.45, pres: 1013, name: "Szczecin"}
  ▶ sys: {type: 2, id: 2034200, country: 'PL', sunrise: 1731652066, sunset: 1731665037, timezone: 3600}
  visibility: 10000
  ▶ weather: [...]
  ▶ wind: {speed: 3.58, deg: 250}
  ▶ [[Prototype]]: Object
```

Punkty:	0	1
---------	---	---

Wstaw zrzut ekranu kodu odpowiedzialnego za wysyłanie żądania do forecast za pomocą Fetch:

AI1 LAB D – Nazwisko Imię – Wersja 1

```
fetch(url) Promise<Response>
  .then(response : Response => {
    if (!response.ok) throw new Error('Nie udało się pobrać prognozy.');
    return response.json();
  }) Promise<any>
  .then(data => {
    console.log("Prognoza pogody na 5dni:", data);
    const forecastList = data.list.slice(0, 5).map(item => {
      const date : Date = new Date(item.dt * 1000);
      const formattedDate : string = `${String(date.getDate()).padStart(2, '0')}/${String(item.main.temp)}°C, ${item.weather[0].description}`;
      return `${formattedDate}: Temp: ${item.main.temp}°C, ${item.weather[0].description}`;
    });
    document.getElementById('forecastInfo').innerText = forecastList.join('\n');
  }) Promise<void>
  .catch(error => alert(error.message));

```

Wstaw zrzut ekranu pokazujący otrzymaną odpowiedź za pomocą `console.log()` w przeglądarce.

```
Prognoza pogody na 5dni: js.js:45
▼ {cod: '200', message: 0, cnt: 40, list: Array(40), city: {...}} ⓘ
  ▼ city:
    ► coord: {lat: 53.4289, lon: 14.553}
      country: "PL"
      id: 3083829
      name: "Szczecin"
      population: 407811
      sunrise: 1731652066
      sunset: 1731683120
      timezone: 3600
    ► [[Prototype]]: Object
    cnt: 40
    cod: "200"
  ► list: (40) [...], [...], [...], [...], [...], [...], [...], [...], [...], [...], [...], [...], [...], [...], [...]
    message: 0
  ► [[Prototype]]: Object
```

Punkty:	0	1
---------	---	---

Wstaw zrzut ekranu przedstawiającego wizualizację prognoz pogody:

Pogodynka

szczecin

sprawdź pogodę

Bieżąca Pogoda:



Temperatura: 6.39°C, Opis: pochmurnie

Prognoza pogody na 5dni:

15/11/2024: Temp: 7.63°C, pochmurnie

15/11/2024: Temp: 8.17°C, zachmurzenie umiarkowane

15/11/2024: Temp: 9.39°C, zachmurzenie duże

15/11/2024: Temp: 9.54°C, zachmurzenie duże

16/11/2024: Temp: 9.21°C, zachmurzenie duże

Upewnij się, że widoczne są pasek wyszukiwania ze wskazaną miejscowością, a także zarówno pogoda bieżąca jak i prognozy pogody.

Punkty:	0	1
---------	---	---

COMMIT PROJEKTU DO GIT

Zacommituj i pushnij swoje rozwiązanie do repozytorium GIT.

Upewnij się, czy wszystko dobrze się wysłało. Jeśli tak, to z poziomu przeglądarki utwórz branch o nazwie lab-d na podstawie głównej gałęzi kodu.

Podaj link do brancha lab-d w swoim repozytorium:

...link, np. <https://github.com/inazwisko/ai1-lab/tree/lab-d...>

PODSUMOWANIE

W kilku zdaniach podsumuj zdobyte podczas tego laboratorium umiejętności.

Dowiedziałam się jak działa API i jak można go używać 😊

...podsumowanie...

Zweryfikuj kompletność sprawozdania. Utwórz PDF i wyślij w terminie.