Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра автоматизованих систем обробки інформації та управління

Лабораторна робота №6

З дисципліни «Проектування інформаційних систем»

На тему: «Використання і створення API»

Виконала студентка гр. ІС-91
Морквіна В.В.
Перевірив доц. каф. АСОІУ
Попенко В. Д.

Київ
2021

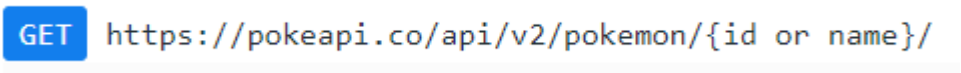
Завдання: Підключити зовнішній сервіс до свого рішення. Сервіс можна використовувати існуючий або створити власний. Приклади зовнішніх сервісів наведені в Додатку.

Вправа 6.1. Обрати зовнішній сервіс і описати сценарій його використання. Створити в GitHub репозиторій з описом сценарію використання API зовнішнього застосування.

PokeApi надає інформацію про покемонів, їх ходи, здібності, типи, групи яєць та багато-багато іншого.

Мною було обрано отримувати дані про покемона за посиланням:

https://pokeapi.co/api/v2/pokemon/squirtle/



Повертає наступну інформацію:

id	The identifier for this resource.	*integer*
name	The name for this resource.	*string*
base_experience	The base experience gained for defeating this Pokémon.	*integer*
height	The height of this Pokémon in decimetres.	*integer*
is_default	Set for exactly one Pokémon used as the default for each species.	*boolean*
order	Order for sorting. Almost national order, except families are grouped together.	*integer*
weight	The weight of this Pokémon in hectograms.	*integer*
abilities	A list of abilities this Pokémon could potentially have.	list *PokemonAbility*
forms	A list of forms this Pokémon can take on.	list *NamedAPIResource (PokemonForm)*
game_indices	A list of game indices relevent to Pokémon item by generation.	list *VersionGameIndex*
held_items	A list of items this Pokémon may be holding when encountered.	list *PokemonHeldItem*
location_area_encounters	A link to a list of location areas, as well as encounter details pertaining to specific versions.	*string*
moves	A list of moves along with learn methods and level details pertaining to specific version groups.	list *PokemonMove*
sprites	A set of sprites used to depict this Pokémon in the game. A visual representation of the various sprites can be found at PokeAPI/sprites	PokemonSprites
species	The species this Pokémon belongs to.	*NamedAPIResource (PokemonSpecies)*
stats	A list of base stat values for this Pokémon.	list *PokemonStat*
types	A list of details showing types this Pokémon has.	list *PokemonType*


```
id: 12
name: "butterfree"
base_experience: 178
height: 11
is_default: true
order: 16
weight: 320
▶ abilities: [] 1 item
▶ forms: [] 1 item
▶ game_indices: [] 1 item
▶ held_items: [] 1 item
location_area_encounters: "https://pokeapi.co/api/v2/pokemon/12/encounters"
▶ moves: [] 1 item
▶ species: {} 2 keys
▶ sprites: {} 10 keys
▶ stats: [] 1 item
▶ types: [] 1 item
```

Дані надаються в наступному вигляді:

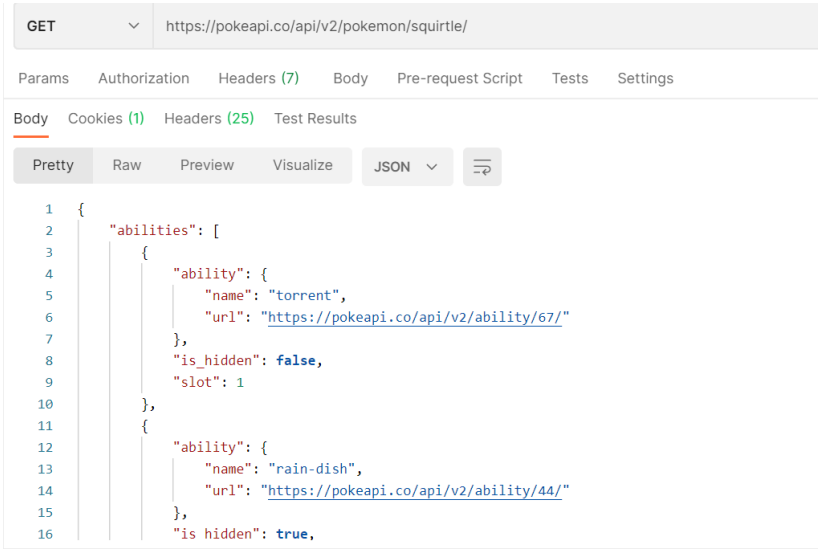


Рисунок 1. Перевірка роботи API за допомогою Postman

Вправа 6.2. Реалізувати на будь-якій мові програмування виклик API зовнішнього застосування і візуалізацію відповіді. Додати в GitHub код і опис прикладу застосування API.

Відображення даних отриманих з API:

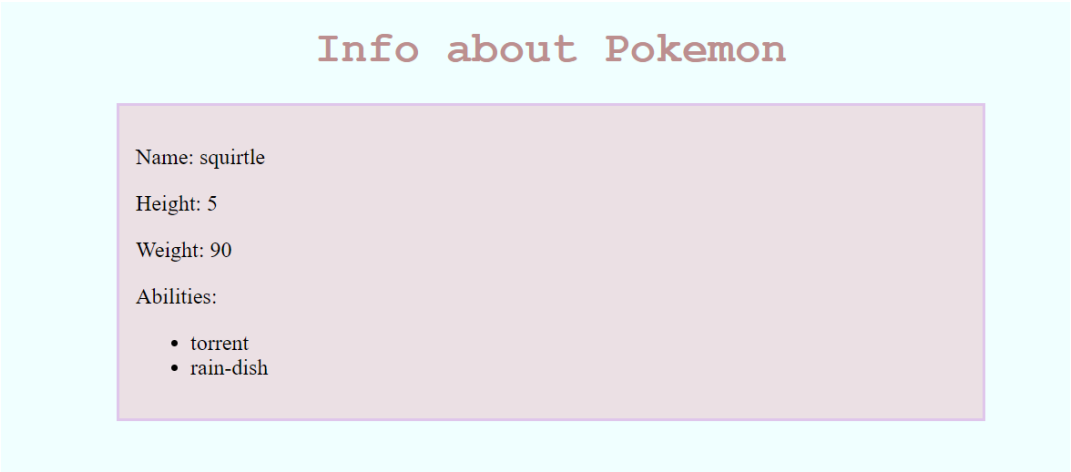


Рис 2. Відображення даних отриманих з API

Таким чином, можна відіслати запит до API і подивитись відповідь у браузері.

В HTML документі із зовнішнім кодом (у окремому файлі) JavaScript виклик виглядає так:

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="stylesheet" href="style.css" />
    <script src="script.js"></script>
    <title>Info about Pokemon</title>
  </head>
  <body onload="data()">
    <h1>Info about Pokemon</h1>
    <section></section>
  </body>
</html>

```

Рис. 3. Код для відображення даних в HTML документі

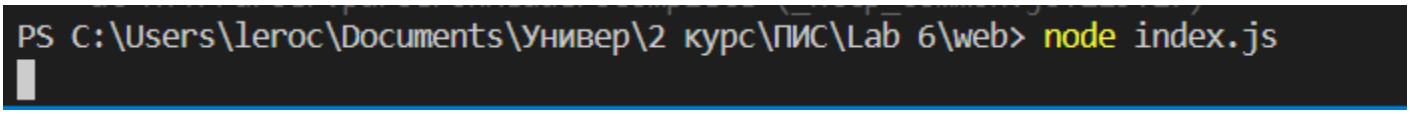
```javascript
function data() {
  let requestURL = 'https://pokeapi.co/api/v2/pokemon/squirtle/';
  let request = new XMLHttpRequest();
  request.open('GET', requestURL);
  request.responseType = 'json';
  request.send();
  request.onload = function () {
    var quote_json = request.response;
    showPokemon(quote_json);
  };
  function showPokemon(jsonObj) {
    let section = document.querySelector('section');
    let block1 = document.createElement('div');
    let name = document.createElement('p');
    name.textContent = 'Name: ' + jsonObj.name;
    let height = document.createElement('p');
    height.textContent = 'Height: ' + jsonObj.height;
    let weight = document.createElement('p');
    weight.textContent = 'Weight: ' + jsonObj.weight;
    let pAbility = document.createElement('p');
    pAbility.textContent = 'Abilities: ';
    let abilities = document.createElement('ul');
    for (let i in jsonObj.abilities) {
      let ability = document.createElement('li');
      ability.textContent = jsonObj.abilities[i].ability.name;
      abilities.append(ability);
    }
    section.append(block1);
    block1.append(name);
    block1.append(height);
    block1.append(weight);
    block1.append(pAbility);
    block1.append(abilities);
  }
}

```

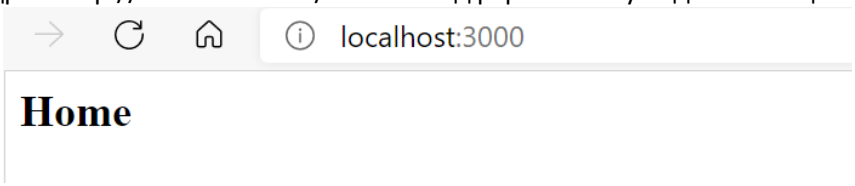
Рис. 4. Код на JS у зовнішньому файлі для відображення даних в HTML документі

Вправа 6.3. Реалізувати на будь-якій мові програмування http-сервер, організувати звертання до нього і отримати відповідь у браузері. Звертання має містити Ваш логін у Moodle. Відповідь має містити Ваші особисті дані (прізвище, ім’я, курс, група).

Сервер створюємо на локальному комп’ютері і доступ до нього здійснюємо через порт 3000. Програмний файл index.js у Windows запускається з консолі Windows.

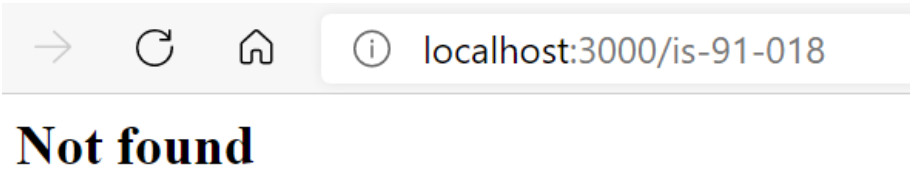


У браузері по адресі http://localhost:3000/ бачимо відформатовану згідно таблиці стилів відповідь сервера.



Реалізуємо звертання, яке буде містити мій логін у Moodle. Відповідь містить мій особисті дані (прізвище, ім’я,курс, група).

Реалізовуємо звертання, яке буде містити те, що не обробляється.



```
const http = require('http');

http
  .createServer(function (req, res) {
    res.setHeader('Content-Type', 'text/html; charset=utf-8;');

    if (req.url === '/home' || req.url === '/') res.write('<h2> Home</h2>');
    else if (req.url === '/is-91-019')
      res.write('<h2> Морквіна Валерія ІС-91, 2 курс</h2>');
    else res.write('<h2>Not found</h2>');

    res.end();
  })
  .listen(3000);

```

Рис 5. Код на JS для побудови http-сервера засобами Node.js

Посилання на репозиторій на GitHub:
morkvinaValeria/is-91-019: ПИС (github.com)