Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №5

з дисципліни <<Технології розробки програмного забезпечення>>

Тема <<Шаблони «ADAPTER», «BUILDER», «COMMAND», «CHAIN OF RESPONSIBILITY», «PROTOTYPE»>>

Виконав:
студент ІА-23
Содолиснький Вадим

Перевірив:
Мягкий М. Ю.

Київ 2024

Тема: Шаблони «ADAPTER», «BUILDER», «COMMAND», «CHAIN OF RESPONSIBILITY», «PROTOTYPE».

Хід роботи

Патерн Адаптер (ADAPTER)

Шаблон «Adapter» використовується для приведення інтерфейсу одного класу у відповідність до вимог іншого інтерфейсу. Це дозволяє об'єктам із несумісними інтерфейсами працювати разом.

Переваги

Сумісність з новими або сторонніми бібліотеками без зміни коду клієнта. Гнучкість — дозволяє використовувати класи з несумісними інтерфейсами. Повторне використання коду — адаптер можна використовувати для різних цілей.

Проблема

Оскільки в мене клієнт-серверна архітектура, то мені потрібно реалізувати взаємодію клієнта з сервером. Я робив це за допомогою класу HttpClient, який надсилає повідомлення у форматі json. Але оскільки мені потрібно надсилати дані у вигляді класів, то потрібно перетворювати об’єкти у рядок json.

Рішення

В цьому випадку я використав патерн адаптер для перетворення інформації, яку потрібно надіслати у формат json. Потім ці дані прочитає сервер і перетворить їх у потрібний клас.

```
public static class JsonRequestConvert
{
    3 references
    public static StringContent ConvertToJsonRequest(object obj)
    {
        return new StringContent(
            System.Text.Json.JsonSerializer.Serialize(obj),
            System.Text.Encoding.UTF8,
            "application/json"
        );
    }
}
```

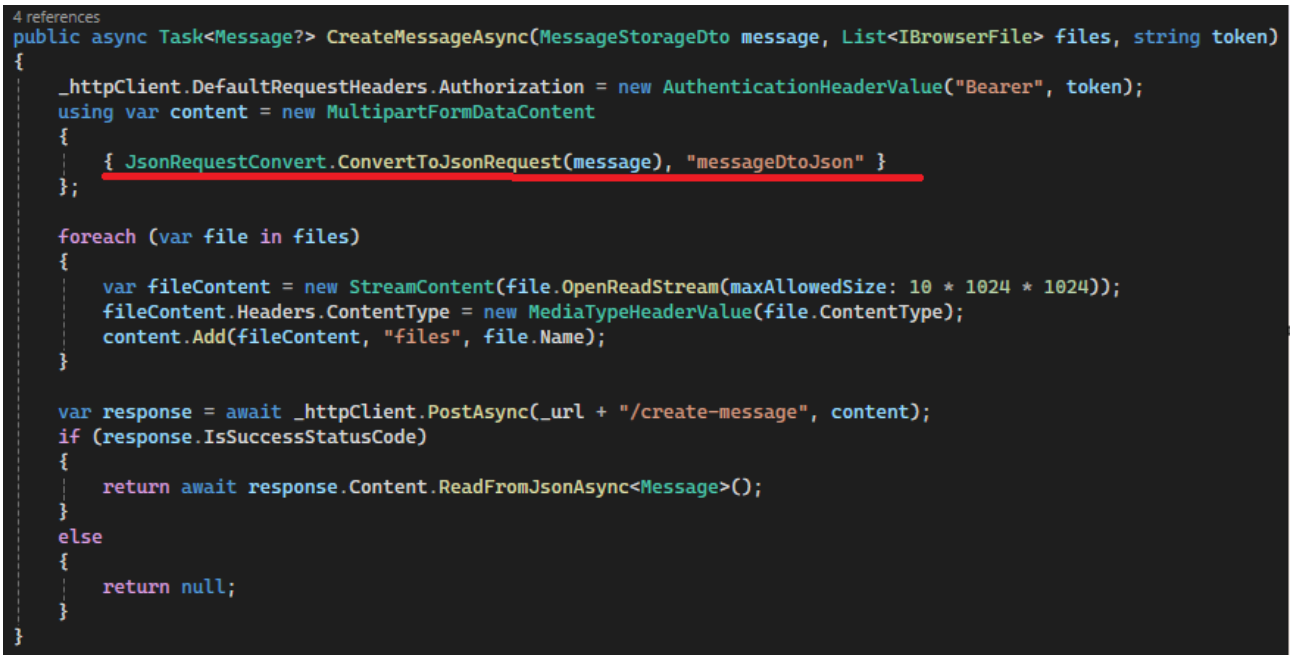
Рис. 1 – Клас, який приймає об’кт та повертає StringContent для надсилання даних до сервера

```
4 references
public async Task<string?> LoginAsync(string username, string password)
{
    var loginRequest = new { Email = username, Password = password };
    var response = await _httpClient.PostAsync(_url + "/login", JsonRequestConvert.ConvertToJsonRequest(loginRequest));
    if (response.IsSuccessStatusCode)
    {
        var result = await response.Content.ReadFromJsonAsync<LoginResponse>();
        return result?.Token;
    }
    else
    {
        var error = await response.Content.ReadAsStringAsync();
        throw new Exception($"Login failed: {error}");
    }
}

0 references
public async Task<string?> RegisterAsync(string username, string password)
{
    var registerRequest = new { Email = username, Password = password };
    var response = await _httpClient.PostAsync(_url + "/signup", JsonRequestConvert.ConvertToJsonRequest(registerRequest));

    if (response.IsSuccessStatusCode)
    {
        var result = await response.Content.ReadFromJsonAsync<LoginResponse>();
        return result?.Token;
    }
    else
    {
        var error = await response.Content.ReadAsStringAsync();
        throw new Exception($"Registration failed: {error}");
    }
}
```

Рис. 2 – Використання JsonRequestConvert у API



```
4 references
public async Task<Message?> CreateMessageAsync(MessageStorageDto message, List<IBrowserFile> files, string token)
{
    _httpClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", token);
    using var content = new MultipartFormDataContent
    {
        { JsonRequestConvert.ConvertToJsonRequest(message), "messageDtoJson" }
    };

    foreach (var file in files)
    {
        var fileContent = new StreamContent(file.OpenReadStream(maxAllowedSize: 10 * 1024 * 1024));
        fileContent.Headers.ContentType = new MediaTypeHeaderValue(file.ContentType);
        content.Add(fileContent, "files", file.Name);
    }

    var response = await _httpClient.PostAsync(_url + "/create-message", content);
    if (response.IsSuccessStatusCode)
    {
        return await response.Content.ReadFromJsonAsync<Message>();
    }
    else
    {
        return null;
    }
}
```

Рис. 3 – Використання JsonRequestConvert у API

Висновок: я використав патерн Adapter для перетворення об’єктів у формат json для надсилання на сервер.