Практическая работа №22
Выполнение HTTP-запросов методами асинхронного программирования

1Цель работы

1.1 Научиться реализовывать и запускать асинхронные операции на C#.
1.2 Научиться выполнять HTTP-запросы, используя асинхронные операции на C#.

2 Литература

2.1 https://docs.microsoft.com/ru-ru/dotnet/csharp/programming-guide/concepts/async/
2.2 https://docs.microsoft.com/ru-ru/dotnet/api/system.net.http.httpclient?view=net-5.0
2.2 https://metanit.com/sharp/tutorial/ - гл.15

3 Задание

В качестве источника json-данных использовать сайт https://jsonplaceholder.typicode.com/

3.1 Реализация чтения данных по сети

3.1.1 Создать в консольном приложении метод, выполняющий в асинхронном режиме чтение данных с определенной веб-страницы. Адрес страницы должен указываться пользователем и передаваться в параметрах метода. Метод должен возвращать результат в виде строки.

Для получения данных в виде строки можно использовать асинхронный метод GetStringAsync класса HttpClient:

```
// client – поле класса
private static readonly HttpClient client = new HttpClient();

// response – переменная, используется в методе чтения данных с веб-страницы
string response = await client.GetStringAsync(адрес);
```

3.1.2 Полученный результат вывести на экран в следующем формате:
адрес страницы:
текст страницы / текст ошибки

3.2 Реализация записи в файл полученных по сети данных

3.2.1 Создать в консольном приложении метод, выполняющий в асинхронном режиме сохранение в текстовый файл данных с определенной веб-страницы. Адрес страницы и имя файла должны указываться пользователем и передаваться в параметрах метода.

Проинформировать пользователя о том, что:
- страницу «адрес страницы» удалось загрузить,
- файл «имя файла» удалось сохранить.

3.2.2 Предоставить пользователю возможность неоднократного указания адресов страниц и файлов (т.е. параллельно может выпоняться чтение и запись нескольких файлов).

3.3 Загрузка данных в оконном приложении

3.3.1 Создать в оконном приложении метод, выполняющий в асинхронном режиме чтение данных с определенной веб-страницы. Адрес страницы должен передаваться в параметрах метода. Метод должен возвращать результат в виде строки.

3.3.2 Добавить на форму поле ввода адреса страницы, кнопку «Загрузить», PictureBox (по умолчанию не видим) и RichTextBox.

Вызвать созданный асинхронный метод в асинхронном обработчике нажатия на кнопку «Загрузить». После того, как пользователь нажал на кнопку:

- кнопка должна блокироваться,
- сделать pictureBox видимым и загрузить в него в асинхронном режиме gif для отображения процесса загрузки:

```
имя pictureBox.LoadAsync(имя gif файла);
```

- вызвать метод чтения данных по сети и отобразить результат в richTextBox,
- сделать pictureBox невидимым,
- кнопка должна сдать разблокироваться.

3.4 Сохранение файлов в оконном приложении

3.4.1 Создать в оконном приложении метод, выполняющий в асинхронном режиме запись в файл данных с определенной веб-страницы. Адрес страницы и имя файла должны передаваться в параметрах метода.

3.4.2 Добавить на форму кнопку «Сохранить» и поля ввода для указания пользователем данных.

Вызвать созданный асинхронный метод в асинхронном обработчике нажатия на кнопку «Сохранить». После того, как пользователь нажал на кнопку:

- вызвать метод записи данных,
- проинформировать пользователя о результате выполнения операции.

3.5 Отправка данных на сервер

3.5.1 Создать и проверить в оконном приложении метод, выполняющий в асинхронном режиме отправку json-данных на страницу https://jsonplaceholder.typicode.com/posts методом POST.

Метод должен возвращать результат выполнения запроса в виде строки.

В процессе загрузки на форме должен отображаться лоадер в pictureBox.

Для отправки данных можно использовать следующий код:

```
HttpContent content = new StringContent("отправляемые данные", Encoding.UTF8,
    "application/json");
HttpResponseMessage result = await client.PostAsync(адрес, content);
if (result.IsSuccessStatusCode) // если данные удалось отправить
{
    // 1 вариант: полученный статус, например, Created
    response = result.StatusCode.ToString();
    // 2 вариант: полученный текст
    response = await result.Content.ReadAsStringAsync();
}
```

Для проверки:
- можно передать произвольные json-данные, например пустой набор: {}
- использовать адрес https://jsonplaceholder.typicode.com/posts
- сравнить полученный текст результата со следующим (корректным):

```
{
  id: 101,
  title: 'foo',
  body: 'bar',
  userId: 1
}
```

4 Порядок выполнения работы

4.1 Выполнить все задания из п.3 в одном решении PractWork22. Возможные ошибки требуется обрабатывать. Выполнить форматирование и рефакторинг кода.

4.2 Ответить на контрольные вопросы.

5 Содержание отчета

5.1 Титульный лист
5.2 Цель работы
5.3 Ответы на контрольные вопросы
5.4 Вывод

6 Контрольные вопросы

6.1 Для чего используется класс HttpClient?
6.2 Для чего используется класс HttpContent?
6.3 Для чего используется класс HttpResponseMessage?