Лабораторная работа № 8.

Тема: Windows Forms. Класс HttpClient.

Цель: закрепить теоретические знания и применить их на практике.

Задание:

Вариант №1.

Задание 1. Загрузка текста из интернета в TextBox.
Создайте форму с текстовым полем (TextBox) и кнопкой "Загрузить". При нажатии кнопки асинхронно загрузите содержимое текстового файла по URL (например, https://www.gutenberg.org/files/11/11-0.txt) и отобразите его в TextBox.
Подсказки:
Используйте класс HttpClient для выполнения HTTP-запросов:

```
private async void LoadButton_Click(object sender, EventArgs e)
{
    using (HttpClient client = new HttpClient())
    {
        string content = await client.GetStringAsync("https://example.com/sample.txt");
        textBox.Text = content;
    }
}
```

Убедитесь, что UI остается отзывчивым во время загрузки данных.

Задание 2. Загрузка изображения в PictureBox.
Создайте форму с элементом PictureBox и кнопкой "Загрузить изображение". При нажатии кнопки асинхронно загрузите изображение по URL (например, https://cataas.com/cat) и отобразите его в PictureBox.
Подсказки:

Используйте метод GetByteArrayAsync для загрузки изображения в виде массива байтов:

```
private async void LoadImageButton_Click(object sender, EventArgs e)
{
    using (HttpClient client = new HttpClient())
    {
        byte[] imageBytes = await client.GetByteArrayAsync("https://example.com/image.jpg");
        pictureBox.Image = Image.FromStream(new MemoryStream(imageBytes));
    }
}
```

Задание 3. Получение JSON и отображение в DataGridView.
Создайте форму с DataGridView и кнопкой "Загрузить данные". При нажатии кнопки асинхронно загрузите JSON-данные с сервера (например, случайные Факты о кошках) и отобразите их в DataGridView.
Подсказки:
Используйте библиотеку System.Text.Json для десериализации JSON:

```
private async void LoadDataButton_Click(object sender, EventArgs e)
{
    using (HttpClient client = new HttpClient())
    {
        string json = await client.GetStringAsync("https://catfact.ninja/fact");
        var cat = JsonSerializer.Deserialize<Cat>(json);
                    ...
    }
```

```
}

public class Cat
{
    public string fact { get; set; }
}
```

Задание 4. Отправка POST-запроса с данными из TextBox.
Создайте форму с текстовыми полями для ввода имени и должности, а также кнопкой "Отправить". При нажатии кнопки отправьте данные на сервер с помощью POST-запроса. После успешной отправки выведите сообщение об успехе.
Подсказки:
Используйте метод PostAsync для отправки данных.
Используйте библиотеку System.Text.Json для сериализации JSON

```
private async void SendButton_Click(object sender, EventArgs e)
{
    var user = new { name = nameTextBox.Text, job = jobTextBox.Text };
        // Сериализуем объект в JSON
        string json = JsonSerializer.Serialize(user);

        // Создаем контент для отправки (объект HttpContent)
        HttpContent content = new StringContent(json, Encoding.UTF8, "application/json");

    using (HttpClient client = new HttpClient())
    {
        var response = await client.PostAsync("https://reqres.in/api/users", content);
        ...
    }
}
```

Задание 5. Загрузка изображений в FlowLayoutPanel.
Создайте форму с элементом FlowLayoutPanel и кнопкой "Загрузить галерею". При нажатии кнопки асинхронно загрузите несколько изображений по URL и добавьте их в FlowLayoutPanel.
Подсказки:
Создайте PictureBox для каждого изображения и добавьте его в FlowLayoutPanel:

```
private async void LoadGalleryButton_Click(object sender, EventArgs e)
{
    string[] urls = { "https://cataas.com/cat", "https://picsum.photos/300/200" };
    foreach (var url in urls)
    {
        using (HttpClient client = new HttpClient())
        {
            byte[] imageBytes = await client.GetByteArrayAsync(url);
            PictureBox pictureBox = new PictureBox
            {
                Image = Image.FromStream(new MemoryStream(imageBytes)),
                                        ...
            };
            flowLayoutPanel.Controls.Add(pictureBox);
        }
    }
}
```

Вариант 2

Задание 1. Загрузка HTML-страницы и поиск текста.

Создайте форму с текстовым полем (TextBox) и кнопкой "Поиск". При нажатии кнопки асинхронно загрузите HTML-страницу по URL и найдите в ней заданное слово (Например "Rabbit)". Выведите количество найденных совпадений в метку.

Подсказки:

Используйте метод GetStringAsync для загрузки HTML:

```
private async void SearchButton_Click(object sender, EventArgs e)
{
   using (HttpClient client = new HttpClient())
   {
      string html = await client.GetStringAsync("https://www.gutenberg.org/files/11/11-0.txt");
      int count = html.Split(new[] { searchTextBox.Text }, StringSplitOptions.None).Length - 1;
      label.Text = $"Найдено совпадений: {count}";
   }
}
```

Задание 2. Загрузка погодных данных и отображение в Label.

Создайте форму с текстовым полем для ввода названия города, кнопкой "Получить погоду" и метками для отображения температуры, скорости ветра и состояния погоды. При нажатии кнопки загрузите данные о погоде через API (например, weatherapi.com) и отобразите их.

Подсказки:

Используйте API для получения JSON-данных:

```
private async void GetWeatherButton_Click(object sender, EventArgs e)
{
   string token = "5f78743f2cec4634a4345232250403";
        string city = textBox1.Text;
        using (HttpClient client = new HttpClient())
        {
                string json = await
client.GetStringAsync($"https://api.weatherapi.com/v1/current.json?key={token}&q={city}");
                var weather = JsonSerializer.Deserialize<WeatherData>(json);
                label1.Text = $"Температура: {weather.current.temp_c}°C";
                ...
        }
}

public class WeatherData
{
        public Location location { get; set; }
        public Current current { get; set; }
}
public class Location
{
        public string name { get; set; }
}

public class Current
{
        public float temp_c { get; set; }
        ...

}
```

Задание 3. Загрузка изображения из Json.
Создайте форму с элементом PictureBox и кнопкой "Загрузить rfhnbyre". При нажатии кнопки асинхронно отправьте запрос, десериализуйте ответ, по полученному url загрузите картинку в PictureBox.
Подсказки:

```
async void button_Click(object sender, EventArgs e)
{
	// Отправляем GET-запрос к API
	string json = await client.GetStringAsync("https://dog.ceo/api/breeds/image/random");
	var dog = JsonSerializer.Deserialize<Dog>(json);
	label1.Text = dog.message;

	// Загружаем изображение по URL
	byte[] imageBytes = await client.GetByteArrayAsync(dog.message);

	// Преобразуем массив байтов в объект Image
	using (MemoryStream ms = new MemoryStream(imageBytes))
	{
		pictureBox.Image = Image.FromStream(ms);
	}
}

public class Dog
{
	public string message { get; set; }
	public string status { get; set; }

}
```

Задание 4. Загрузка файлов и сохранение на диск.
Создайте форму с текстовым полем для ввода URL файла, кнопкой "Скачать". При нажатии кнопки асинхронно загрузите файл и сохраните его на диск.
Подсказки:
Используйте метод DownloadFileAsync:

```
private async void DownloadButton_Click(object sender, EventArgs e)
{
   using (HttpClient client = new HttpClient())
   {
      byte[] fileBytes = await client.GetByteArrayAsync(urlTextBox.Text);
      File.WriteAllBytes("downloaded_file", fileBytes);
      MessageBox.Show("Файл успешно скачан!");
   }
}
```

Задание 5. Отправка данных формы регистрации.
Создайте форму с полями для ввода имени, email и пароля, а также кнопкой "Зарегистрироваться". При нажатии кнопки отправьте данные на сервер с помощью POST-запроса. После успешной регистрации выведите сообщение об успехе.
Подсказки:
Сериализуйте данные в JSON и отправьте их:

```
private async void button_Click(object sender, EventArgs e)
{
	var user = new { Name = textBox1.Text, Email = textBox2.Text, Password =
textBox3.Text };
	using (HttpClient client = new HttpClient())
	{
		string json = JsonSerializer.Serialize(user);
```

```
                HttpContent content = new StringContent(json, Encoding.UTF8,
"application/json");
                var response = await client.PostAsync("https://reqres.in/api/users", content);
                ...
        }
}
```

Общие подсказки:
HttpClient: Используйте using для автоматического освобождения ресурсов.
Методы GetStringAsync, GetByteArrayAsync, PostAsync упрощают работу с HTTP-запросами.
Асинхронность: Используйте ключевые слова async и await для асинхронных операций.
Не забывайте блокировать элементы управления во время выполнения запросов.
DataGridView и ListBox подходят для отображения списков данных.

Отчет должен содержать (см. образец):
- номер и тему лабораторной работы;
- фамилию, номер группы студента и вариант задания;
- скриншоты окна Visual Studio с исходным кодом программ и комментариями;
- скриншоты с результатами выполнения программ;
- пояснения, если необходимо;
- выводы.

Отчеты в формате **pdf** отправлять на email: **colledge20education23@gmail.com**