1. **ScraperGUI 1.1** - файл для установки Scraper на iOS

2) **PublishWin** – файл для установки Scraper на Windows

3) **ScrapersProject** - файл с начальным заданием без дополнений

**Отчёт по практике: Создание консольных скраперов на C#**  
(для файла ScrapersProject)

# 1. Цель практики

Создать консольные приложения на языке C# (.NET 9), которые:  
1. Получают данные из таблиц финансовых сайтов.  
2. Сохраняют данные в файлы формата TSV с табуляцией.  
3. Ведут логирование и обрабатывают возможные ошибки.  
4. Структурированы по стандартам читаемого и поддерживаемого кода.

# 2. Задачи

1. Изучить основы работы с C# и библиотекой HtmlAgilityPack для парсинга HTML.  
2. Разработать два отдельных консольных приложения:  
 - WienerBorseScraper — для сайта Wiener Börse Bonds.  
 - JseDeltaTradesScraper — для сайта JSE Delta and Structured Option Trades.  
3. Реализовать логирование через log4net.  
4. Сохранять таблицы в TSV с заголовками и разделителями табами.  
5. Добавить обработку исключений для ошибок сети и отсутствия таблиц.  
6. Структурировать код в читаемую и поддерживаемую архитектуру.

# 3. Средства и технологии

- Язык программирования: C# 11 / .NET 9  
- Библиотеки:  
 - HtmlAgilityPack — для парсинга HTML  
 - log4net — для логирования  
- Среда разработки: Visual Studio Code  
- ОС: macOS  
- Дополнительные инструменты:  
 - Chrome Developer Tools (F12) — для анализа структуры таблиц и сетевых запросов  
 - Insomnia — для отладки HTTP-запросов

# 4. Ход работы

## 4.1 Изучение литературы и материалов

1. Книги по C#:  
 - “C# 11 and .NET 9 – Modern Cross-Platform Development”, Марк Дж. Прайс.  
 - “Pro C# 10 with .NET 6/9”, Эндрю Трост.  
2. Документация и онлайн-ресурсы:  
 - HtmlAgilityPack Documentation  
 - log4net Documentation  
3. Практические инструменты:  
 - Использование F12 в Chrome для анализа DOM таблиц.  
 - Проверка и отладка HTTP-запросов через Insomnia.

## 4.2 Разработка структуры проекта

1. Созданы два консольных приложения: WienerBorseScraper, JseDeltaTradesScraper  
2. В каждом проекте создана папка Services/ с классами скрапера и CsvWriterHelper  
3. В корне каждого проекта добавлен log4net.config  
4. Принцип организации:  
 - Каждый скрапер инкапсулирует логику получения HTML и извлечения таблицы.  
 - CsvWriterHelper отвечает только за сохранение данных в файл.  
 - Основной файл Program.cs инициирует логгер и запускает скрапер.

## 4.3 Разработка логики скрапинга

1. Анализ HTML страниц: определены теги table, thead, tbody, th, td.  
2. Получение HTML через HttpClient с обработкой ошибок.  
3. Парсинг таблицы через HtmlAgilityPack, очистка текста HtmlEntity.DeEntitize.  
4. Запись данных в TSV с табуляцией, первая строка — заголовки, имя файла содержит дату.  
5. Логирование каждого этапа работы через log4net.  
6. Обработка исключений: отсутствие таблицы, ошибки сети, общие исключения.

## 4.4 Тестирование

1. Запуск скрапера для Wiener Börse:  
dotnet run --project WienerBorseScraper/WienerBorseScraper.csproj  
Файл: wienerborse\_2025-09-24.tsv  
Лог: wienerborse\_scraper.log  
  
2. Запуск скрапера для JSE:  
dotnet run --project JseDeltaTradesScraper/JseDeltaTradesScraper.csproj  
Файл: jse\_delta\_trades\_2025-09-24.tsv  
Лог: jse\_delta\_trades\_scraper.log

## 4.5 Результаты работы

|  |  |  |
| --- | --- | --- |
| Проект | Файл данных | Лог |
| WienerBorseScraper | wienerborse\_2025-09-24.tsv | wienerborse\_scraper.log |
| JseDeltaTradesScraper | jse\_delta\_trades\_2025-09-24.tsv | jse\_delta\_trades\_scraper.log |

Все таблицы сохранены корректно с заголовками и разделителями таб.  
Логирование зафиксировало все этапы работы скрапера.  
Код структурирован и легко читается.

**5. Выводы**

- Практика позволила освоить работу с C# (.NET 9) для реальных задач — скрапинга финансовых сайтов.  
- Освоены библиотеки HtmlAgilityPack и log4net, а также работа с HTTP-запросами.  
- Проекты имеют читаемую структуру, логирование и обработку исключений.  
- Код готов к расширению: добавление новых сайтов, автоматизация запуска.  
- Практика показала важность анализа HTML страницы и тестирования запросов через браузерные инструменты и Insomnia.



