**Лабораторная работа 6**

**HTTP**

(4 занятия)

**Цели**

1. Познакомиться с протоколом HTTP и инструментами работы с HTTP в .NET.

2. Реализовать в соответствии с заданием классы или классы для доступа к web ресурсам по протоколу HTTP.

**Ход работы**

1. Определиться с характером, источником и способом представления информации (см. задание).

2. Продумать необходимые типы и их интерфейсы.

3. Реализовать необходимые типы.

4. Реализовать тестовый пример с использованием вышеупомянутых типов. Модульные тесты подойдут.

5. Подготовить отчет о лабораторной работе (электронный вариант).

В отчет включить:

* цели;
* описание характера, источника и способа представления информации (см. задание);
* диаграмму разработанных типов и схему их взаимодействия друг с другом и со стандартными типами;
* код реализованных типов.

**Задание**

Предлагается программно:

- запросить данные с какого-нибудь web ресурса,

- получить их,

- обработать и отобразить.

***Задания с пояснением на примерах:***

1. привязка к первой лабораторной работе (иерархия классов). В первой работе были реализованы классы "печатное издание, периодическое издание, книга, журнал, учебник". В четвертой они загружают информацию о книгах, журналах и учебниках (название, авторы, год издания и проч.) с сайта (например, [oz.by](http://oz.by/) или [amazon.com](http://amazon.com/)) и заполняют этой информацией экземпляры соответствующих классов в памяти. Другой пример: сделали классы для описания таблицы Менделеева. Загружают с сайта [chemport.ru](http://chemport.ru/) описания элементов запросом вида <http://chemport.ru/pertable/elinfo.php?el=N>, где вместо N указан атомный номер элемента.

2. произвольная информация. Список новостей с [tut.by](http://tut.by/), названия и цены товаров с [onliner.by](http://onliner.by/) - что угодно.

3. API какой-нибудь платформы. Получить список друзей с [vk.com](http://vk.com/), программно твитнуть, вытащить список постов с [tumblr.com](http://tumblr.com/) - различных API великое множество. Этот вариант - "на десятку", он сложнее предыдущих. Не плагиатить реализацию существующих клиентов!

**Требования:**

* только .NET 4.5;
* данные из интернета. В крайнем случае, если интернета нет, настроить локально IIS и тянуть с него.
* для получения данных обязательно использовать ***System.Net.Http.HttpClient***; надо будет подключить сборку System.Net.Http.dll; не больше одного экземпляра этого класса на запрос;
* обязательно создать свой message handler, наследующий от System.Net.Http.DelegatingHandler. Задача этого обработчика - журналировать в текстовые файлы исходящие запросы и входящие ответы в форме, показанной во вложении (соответственно, request.txt и response.txt). Файлы журнала не должны "затирать" друг друга (т.е. если во время выполнения было сделано N запросов и получено M ответов, то файлов должно быть N+M: N с запросами и M с ответами); хранить файлы между запусками необязательно. Файл запроса должен по имени однозначно сопоставляться с файлом соответствующего ответа. Обработчик должен встраиваться в цепь обработчиков сообщений перед экземпляром обработчика  System.Net.Http.HttpClientHandler (с помощью свойства InnerHandler класса DelegatingHandler и его наследников).

***Задание на "плюс балл"*** - сделать журнилирование ответов неблокирующим (т.е. чтобы журналирующий обработчик не ждал получения ответа, а записывал его по мере готовности, не мешая при этом делать новые запросы);

* при реализации сосредоточиться именно на запросе данных, а не на их разборе и отображении.

**Задание повышенной сложности.**

Заключается задание в том, чтобы оформлять журнал запросов-ответов не в виде текстовых файлов, а в виде файлов .saz - сохраненных сессий [Fiddler](http://www.telerik.com/fiddler). Это обыкновенные ZIP-архивы, внутри которых запросы и ответы хранятся в том же текстовом виде плюс метаданные.