**Класс WebClient**

[C# и .NET](https://professorweb.ru/my/csharp/charp_theory/level1/infocsharp.php) --- [Сетевое программирование](https://professorweb.ru/my/csharp/web/level1/web_index.php) --- Класс WebClient

Если необходимо только запросить файл с определенного URI (Uniform Resource Identifier — унифицированный идентификатор ресурса), то простейшим в использовании классом .NET, который подходит для этого, будет **System.Net.WebClient**. Этот исключительно высокоуровневый класс предназначен для выполнения базовых операций с помощью всего одной или двух команд. В настоящее время в .NET Framework поддерживаются URI, начинающиеся с идентификаторов *http:*, *https:* и *file:*.

Важно отметить, что термин URL (Uniform Resource Locator — универсальный локатор ресурсов) больше не используется в новых технических спецификациях, а вместо него отдается предпочтение URI. URI имеет приблизительно тот же смысл, что и URL, но немного более общий, потому что в URL не подразумевается обязательное применение одного из знакомых протоколов, таких как HTTP или FTP.

**Загрузка файлов**

Для загрузки файлов с использованием WebClient доступны два метода. Выбор метода зависит от того, как должно обрабатываться содержимое файла. Если необходимо просто сохранить файл на диске, следует применять **метод DownloadFile()**. Этот метод принимает два параметра: URI файла и местоположение (путь и имя файла) для сохранения запрошенных данных:

WebClient Client = new WebClient();

Client.DownloadFile("http://www.professorweb.ru/", "index.php");

Часто приложение должно обрабатывать данные, извлеченные с веб-сайта. Это обеспечивает **метод OpenRead()**, возвращающий ссылку на Stream, которую можно использовать для извлечения данных в память:

WebClient Client = new WebClient();

Stream str = Client.OpenRead("http://www.professorweb.ru/");

В следующем примере демонстрируется применение метода WebClient.OpenRead(). Содержимое загруженной страницы будет отображено в элементе управления TextBox. Для начала создайте новый проект как стандартное приложение WPF и добавьте элемент управления TextBox по имени txb. В начало файла к списку директив using потребуется добавить ссылки на пространства имен System.Net и System.IO. Затем добавьте обработчик клика по кнопке:

private void Button\_Click(object sender, RoutedEventArgs e)

{

WebClient client = new WebClient();

Stream stream = client.OpenRead("http://www.professorweb.ru");

StreamReader sr = new StreamReader(stream);

string newLine;

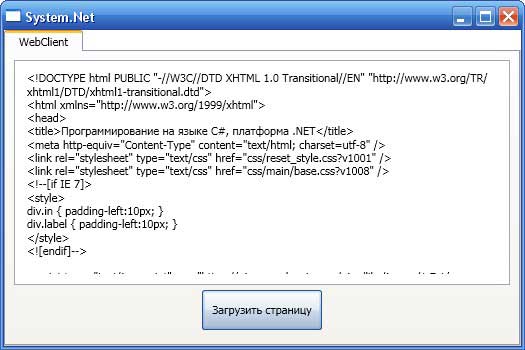
while ((newLine = sr.ReadLine()) != null)

txb.Text += newLine;

stream.Close();

}

В этом примере класс StreamReader из пространства имен System.IO подключается к сетевому потоку. Это позволяет получить данные из потока в виде текста, используя высокоуровневые методы вроде ReadLine(). На рис. показаны результаты запуска этого кода:



Класс WebClient также включает в себя **метод OpenWrite()**. Этот метод возвращает записываемый поток для отправки данных по определенному URI. При этом можно указать метод, который должен использоваться для отправки данных на хост, по умолчанию это POST.

**Выгрузка файлов**

В классе WebClient также имеются методы **UploadFile()** и **UploadData()**. Они используются, когда нужно отправить HTML-форму или выгрузить на сервер целый файл. UploadFile() выгружает файл в указанное местоположение по указанному имени локального файла, в то время как UploadData() выгружает двоичные данные, представленные в виде массива байт, по указанному URI (есть также *метод DownloadData()*, предназначенный для извлечения массива байтов из URI).

WebClient client = new WebClient();

client.UploadFile("http://professorweb.ru/NewFile.html", "samplefile.txt");

byte[] image;

client.UploadData("http://professorweb.ru/NewFile.jpg", image);

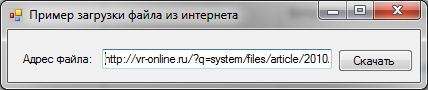
Хотя класс WebClient очень прост в использовании, он обладает весьма ограниченными возможностями. В частности, невозможно передавать регистрационные данные для аутентификации — одна из проблем, связанных с загрузкой данных, состоит в том, что лишь относительно немного сайтов принимают файлы без аутентификации.

В запросы можно добавлять информацию заголовка и проверять заголовки в ответе, но лишь в очень общем смысле — специфическая поддержка какого-то одного протокола отсутствует. Причина в том, что WebClient — класс общего назначения, предназначенный для работы с любым протоколом, позволяющем отправлять запросы и получать ответы (вроде HTTP и FTP). Он не может обработать никаких средств, специфичных для какого-то одного протокола, например, сокеты, которые специфичны для HTTP. Чтобы воспользоваться преимуществами этих средств, нужно работать с семейством классов, основанным на двух других классах из пространства имен System.Net, а именно — WebRequest и WebResponse.

**Скачать файл из Интернета средствами C#**

В C# это сделать проще простого, и я решил не откладывать, а сразу помочь человеку. Открыл Visual Studio и за пять секунд набросал простенький примерчик. Отправил и подумал: «А ведь наверняка, кто-то может столкнуться с аналогичным вопросом?». И решил я сразу помочь всем в виде этой небольшой заметки. Надеюсь, что она кому-нибудь пригодится.

Я уже сказал, что пример крайне прост в написании и требует для реализации чертовски мало времени. Итак, первое, что я сделал – набросал дизайн приложения:



На форме у меня всего лишь одна строка ввода и кнопка «скачать». Собственно по нажатию кнопки и будет происходить загрузка.

Делай форму и переходи в редактор кода. Сразу же подключи два неймспейса:

[code=c#]  
using System.Net;  
using System.IO;  
[/code=c#]

Первый позволит нам работать с сетью, а второй с файлами. Сделав это, создавай обработчик события Click() для единственной кнопки и напиши в нем следующий код:

[code=c#]  
WebClient myWebClient = new WebClient();  
string downloadFileName = System.IO.Path.GetFileName(textBox1.Text);

myWebClient.DownloadFile(textBox1.Text, @"D:\" + downloadFileName);  
[/code=c#]

В первой строчке кода я создаю экземпляр объекта WebClient(). С его помощью мы и будем выполнять загрузку файла из сети. Во второй, я объявляю переменную downloadFileName и присваиваю ей в качестве значения имя загружаемого файла.

Имя файла я получаю при помощи метода GetFileName класса Path. На этом подготовка завершена и можно выполнять саму загрузку файла. Делается она при помощи метода DownloadFile класса WebClient. Методу требуется передать два параметра:

1. Адрес загружаемого файла  
2. Путь, по которому файл будет сохранен.

В первом параметре я указываю текст, который был введен в textBox1, а во втором жестко прописываю D:\ + имя загружаемого файла. Тем самым, все файлы будут сохранятся в корень диска D:.

Попробуй запустить программу, заполнить поле «Адрес файла» и нажть кнопку «Скачать». Через некоторое время (все зависит от размера загружаемого файла и скорости твоего инета) ты увидишь скачанный файлик.