Оглавление

[Оглавление 1](#_Toc173627255)

[Основы работы с сетями в C# и .NET 2](#_Toc173627256)

[Введение в сети и протоколы 2](#_Toc173627257)

[Адреса в .NET 2](#_Toc173627258)

[IPAddress 2](#_Toc173627259)

[Встроенные адреса 3](#_Toc173627260)

[Схема адресации 3](#_Toc173627261)

[Конечная точка IPEndpoint 4](#_Toc173627262)

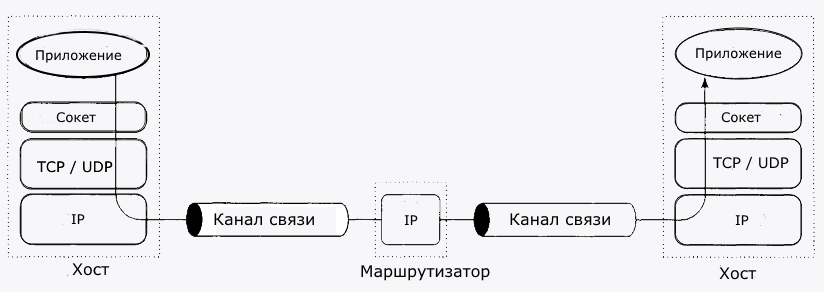
[Адреса Uri 5](#_Toc173627263)

[System.Uri 5](#_Toc173627264)

[UriBuilder 7](#_Toc173627265)

Основы работы с сетями в C# и .NET

## Введение в сети и протоколы



**Основная функциональность фреймворка .NET по работе с сетями содержится в пакете System.Net. Также есть дополнительные пакеты:**

* **System.Net.Http: содержит функциональность по работе с протоколом HTTP**
* **System.Net.NetworkInformation: редоставляет доступ к данным о сетевом трафике и сетевых адресах, а также к прочей информации о хостах сети. Также предоставляет функциональность ping**
* **System.Net.Security: предоставляет сетевые потоки для безопасной связи между хостами**
* **System.Net.Sockets: предоставляет доступ к функциональности сокетов операционной системы**
* **System.Net.WebSockets: предоставляет доступ к реализации инфтерфейса WebSocket**
* **System.Net.Quic: содержит типы, которые реализуют протокол QUIC в соответствии со спецификацией RFC 9000.**

## Адреса в .NET

### IPAddress

**В системе классов .NET ip-адрес представлен классом IPAddress из пространства System.Net.**

**Конструкторы:**

**public IPAddress (byte[] address);**

**public IPAddress (long newAddress);**

**Пример:**

**using System.Net;**

**IPAddress ip = new IPAddress(new byte[] { 127, 0, 0, 1 });**

**// или**

**IPAddress ip = new IPAddress(0x0100007F);**

**// 127.0.0.1, где 1 - 01, 0 - 00, 0 - 00, 127 - 7F**

**// расположены наоборот**

**// или через Parse**

**IPAddress ip = IPAddress.Parse("127.0.0.1");**

**// или через TryParse**

**IPAddress.TryParse("127.0.0.1", out IPAddress? ip);**

**Console.WriteLine(ip); // 127.0.0.1**

### Встроенные адреса

**IPAddress предоставляет ряд адресов по умолчанию через ряд статических свойств:**

* **Статическое свойство Loopback: возвращает объект IPAddress для адреса 127.0.0.1**
* **Статическое свойство Any: возвращает объект IPAddress для адреса 0.0.0.0**
* **Статическое свойство Broadcast: возвращает объект IPAddress для адреса 255.255.255.255**

**IPAddress anyIp = IPAddress.Any;**

**IPAddress localIp = IPAddress.Loopback;**

**IPAddress broadcastIp = IPAddress.Broadcast;**

### Схема адресации

**Свойство класса IPAddress AddressFamily может принимать значения:**

* **AppleTalk: адрес AppleTalk**
* **Atm: адрес собственных служб ATM**
* **Banyan: адрес Banyan**
* **Ccitt: адреса протоколов CCITT, таких как протокол X25**
* **Chaos: адрес протоколов MIT CHAOS**
* **Cluster: адрес кластерных продуктов корпорации Майкрософт**
* **ControllerAreaNetwork: сетевой адрес области контроллера**
* **DataKit: адрес протоколов Datakit**
* **DataLink: адрес интерфейса прямого канала передачи данных**
* **DecNet: адрес DECnet**
* **Ecma: адрес ЕСМА (European Computer Manufacturers Association — европейская ассоциация производителей компьютеров)**
* **FireFox: адрес FireFox**
* **HyperChannel: адрес NSC Hyperchannel**
* **Ieee12844: адрес рабочей группы IEEE 12844**
* **ImpLink: адрес ARPANET IMP**
* **InterNetwork: IPv4-адрес**
* **InterNetworkV6: IPv6-адрес**
* **Ipx: IPX- или SPX-адрес**
* **Irda: IrDA-адрес**
* **Iso: адрес протоколов ISO**
* **Lat: LAT-адрес**
* **Max: MAX-адрес**
* **NetBios: адрес NetBios**
* **NetworkDesigners: адрес шлюзовых протоколов Network Designers OSI**
* **NS: адрес протоколов Xerox NS**
* **Osi: адрес протоколов OSI**
* **Packet: адрес пакета нижнего уровня**
* **Pup: адрес протоколов PUP**
* **Sna: адрес IBM SNA**
* **Unix: локальный адрес Unix для узла**
* **Unknown: семейство неизвестных адресов**
* **Unspecified: семейство неуказанных адресов**
* **VoiceView: адрес VoiceView**

**Пример:**

**IPAddress ip = new IPAddress(new byte[] { 127, 0, 0, 1 });**

**Console.WriteLine(ip.AddressFamily); // InterNetwork**

### Конечная точка IPEndpoint

**Конечная точка представляет объединение IP-адреса и порта и в .NET представляет класс IPEndPoint из пространства имен System.Net.**

**Конструкторы:**

**public IPEndPoint (long address, int port);**

**public IPEndPoint (IPAddress address, int port);**

**Пример:**

**using System.Net;**

**IPAddress ip = IPAddress.Parse("127.0.0.1");**

**IPEndPoint endpoint = new IPEndPoint(ip, 8080);**

**// или**

**IPEndPoint endPoint = IPEndPoint.Parse("127.0.0.1:8080");**

**// или**

**IPEndPoint.TryParse("127.0.0.1:8080", out IPEndPoint? endPoint);**

**Console.WriteLine(endpoint); // 127.0.0.1:8080**

**Свойства IPEndPoint предоставляют доступ к информации о конечной точке:**

* **Address: возвращает или устанавливает IP-адрес**
* **AddressFamily: возвращает схему адресации, которую применяет IP-адрес**
* **Port: возвращает или устанавливает номер порта**

**IPEndPoint endpoint = IPEndPoint.Parse("127.0.0.1:8080");**

**Console.WriteLine(endpoint.Address); // 127.0.0.1**

**Console.WriteLine(endpoint.Port); // 8080**

## Адреса Uri

**Uniform Resource Identifier (URI) - Единый идентификатор ресурса**

**Uniform Resource Locator (URL) - Единый локатор ресурсов - подвид URI**

**Общая форма адреса URL:**

**//[authority/]path[?query][#fragment]**

**authority включает домен и порт, а также возможные учетные данные пользователя как логин и пароль:**

**//[access\_credentials][@]host\_domain[:port]**

**Символ @ отделяет учетные данные от домена. Параметр access\_credentials, который описывает учетные данные, задается в виде**

**[user\_id][:][password]**

**Параметр user\_id предоставляет логин, а password - пароль для доступа к ресурсу. Они отделяются друг от друга двоеточием.**

**Компонент query имеет следующую форму:**

**?[parameter1=value2][(;|&)parameter2=value2]...**

**В конце после символа решетки # может указываться фрагмент - необязательная строка для идентификации компонентов внутри URL. Обычно применяется в веб-браузерах для навигации по частям веб-страницы.**

**При отправке сообщения по URI компьютер обращается к DNS (Domain Name System - система доменных имен), которая сопоставляет URI и IP.**

### System.Uri

**Uri uri = new("http://website.com/");**

**// или**

**Uri uri = new Uri("http://website.com/", UriKind.Absolute);**

**// абсолютный адрес**

**Uri uri = new Uri("sharp/net", UriKind.Relative);**

**// относительный адрес**

**Uri uri = new Uri("http://website.com/sharp/net",**

**UriKind.RelativeOrAbsolute);**

**// абсолютный или относительный**

**// или**

**Uri.TryCreate(string url,**

**UriCreationOptions options, // new UriCreationOptions()**

**out Uri? newUri);**

**Базовые свойства:**

* **AbsoluteUri: возвращает абсолютный адрес URI**

**https://user:password@www.somesite.com/home/index?q1=v1&q2=v2#fragmentName**

* **Authority: возвращает либо имя хоста в соответствии с системой доменных имен DNS, либо IP-адрес и порт сервера.**

**www.somesite.com**

* **Fragment: возвращает фрагмент адреса URI.**

**#fragmentName**

* **Host: возвращает хост.**

**www.somesite.com**

* **IsAbsoluteUri: возвращает true, если адрес абсолютный.**
* **IsDefaultPort: возвращает true, если адрес URI использует порт по умолчанию для своей схемы.**
* **IsFile: возвращает true, если адрес Uri представляет адрес файла.**
* **IsLoopback: возвращает true, если адрес Uri указывает на локальный хост.**
* **OriginalString: возвращает оригинальную строку адреса URI, которая передана в конструктор Uri.**
* **PathAndQuery: возвращает значения свойств AbsolutePath и Query, разделяя их вопросительным знаком (?).**

**/home/index?q1=v1&q2=v2**

* **Port: возвращает номер порта для текущего адреса URI.**

**443**

* **Query: возвращает строку запроса из текущего адреса URI.**

**?q1=v1&q2=v2**

* **Scheme: возвращает схему текущего адреса URI.**

**https**

* **Segments: возвращает массив сегментов пути для текущего адреса URI. Каждый сегмент представляет часть пути, которая ограничена слешами**

**["/", "home/", "index" ]**

* **UserInfo: возвращает имя и пароль пользователя.**

**user:password**

### UriBuilder

**Также для создания адреса можно использовать System.UriBuilder.**

**Свойства UriBuilder:**

* **Uri: возвращает созданный экземпляр Uri.**

**https://user:password@www.somesite.com/home/index?q1=v1&q2=v2#fragmentName**

* **Fragment: возвращает или задает фрагмент URI.**

**#fragmentName**

* **Host: возвращает или задает имя хоста или IP-адрес сервера.**

**www.somesite.com**

* **Password: возвращает или задает пароль.**

**password**

* **Path: возвращает или задает путь к ресурсу, на который ссылается URI.**

**/home/index**

* **Port: возвращает или задает номер порта URI.**

**443**

* **Query: возвращает или задает строку запроса.**

**?q1=v1&q2=v2**

* **Scheme: возвращает или задает схему URI.**

**https**

* **UserName: возвращает или задает имя пользователя.**

**user**

**public UriBuilder (string? scheme,**

**string? host,**

**int port,**

**string? path,**

**string? extraValue)**