

NAME

IO::Socket::INET - Object interface for AF_INET domain sockets

SYNOPSIS

```
use IO::Socket::INET;
```

DESCRIPTION

IO::Socket::INET provides an object interface to creating and using sockets in the AF_INET domain. It is built upon the IO::Socket interface and inherits all the methods defined by IO::Socket.

CONSTRUCTOR

`new ([ARGS])`

Creates an IO::Socket::INET object, which is a reference to a newly created symbol (see the Symbol package). `new` optionally takes arguments, these arguments are in key-value pairs.

In addition to the key-value pairs accepted by IO::Socket, IO::Socket::INET provides.

PeerAddr	Remote host address	<hostname>[:<port>]
PeerHost	Synonym for PeerAddr	
PeerPort	Remote port or service	<service>[(<no>)] <no>
LocalAddr	Local host bind address	hostname[:port]
LocalHost	Synonym for LocalAddr	
LocalPort	Local host bind port	<service>[(<no>)] <no>
Proto	Protocol name (or number)	"tcp" "udp" ...
Type	Socket type	SOCK_STREAM SOCK_DGRAM ...
Listen	Queue size for listen	
ReuseAddr	Set SO_REUSEADDR before binding	
Reuse	Set SO_REUSEADDR before binding (deprecated, prefer ReuseAddr)	
ReusePort	Set SO_REUSEPORT before binding	
Broadcast	Set SO_BROADCAST before binding	
Timeout	Timeout value for various operations	
MultiHomed	Try all addresses for multi-homed hosts	
Blocking	Determine if connection will be blocking mode	

If `Listen` is defined then a listen socket is created, else if the socket type, which is derived from the protocol, is SOCK_STREAM then `connect()` is called.

Although it is not illegal, the use of `MultiHomed` on a socket which is in non-blocking mode is of little use. This is because the first `connect` will never fail with a timeout as the `connect` call will not block.

The `PeerAddr` can be a hostname or the IP-address on the "xx.xx.xx.xx" form. The `PeerPort` can be a number or a symbolic service name. The service name might be followed by a number in parenthesis which is used if the service is not known by the system. The `PeerPort` specification can also be embedded in the `PeerAddr` by preceding it with a ":".

If `Proto` is not given and you specify a symbolic `PeerPort` port, then the constructor will try to derive `Proto` from the service name. As a last resort `Proto` "tcp" is assumed. The `Type` parameter will be deduced from `Proto` if not specified.

If the constructor is only passed a single argument, it is assumed to be a `PeerAddr` specification.

If `Blocking` is set to 0, the connection will be in nonblocking mode. If not specified it defaults to 1 (blocking mode).

Examples:

```
$sock = IO::Socket::INET->new(PeerAddr => 'www.perl.org',
```

```
PeerPort => 'http(80)',
Proto    => 'tcp');

$sock = IO::Socket::INET->new(PeerAddr => 'localhost:smtp(25)');

$sock = IO::Socket::INET->new(Listen    => 5,
                              LocalAddr => 'localhost',
                              LocalPort => 9000,
                              Proto     => 'tcp');

$sock = IO::Socket::INET->new('127.0.0.1:25');

$sock = IO::Socket::INET->new(
    PeerPort => 9999,
    PeerAddr => inet_ntoa(INADDR_BROADCAST),
    Proto    => udp,
    LocalAddr => 'localhost',
    Broadcast => 1 )
    or die "Can't bind : $@\n";
```

NOTE NOTE

As of VERSION 1.18 all IO::Socket objects have autoflush turned on by default. This was not the case with earlier releases.

NOTE NOTE

METHODS

sockaddr ()

Return the address part of the sockaddr structure for the socket

sockport ()

Return the port number that the socket is using on the local host

sockhost ()

Return the address part of the sockaddr structure for the socket in a text form xx.xx.xx.xx

peeraddr ()

Return the address part of the sockaddr structure for the socket on the peer host

peerport ()

Return the port number for the socket on the peer host.

peerhost ()

Return the address part of the sockaddr structure for the socket on the peer host in a text form xx.xx.xx.xx

SEE ALSO

Socket, *IO::Socket*

AUTHOR

Graham Barr. Currently maintained by the Perl Porters. Please report all bugs to <perlbug@perl.org>.

COPYRIGHT

Copyright (c) 1996-8 Graham Barr <gbarr@pobox.com>. All rights reserved. This program is free software; you can redistribute it and/or modify it under the same terms as Perl itself.

