SFML

* [Main Page](http://docs.google.com/index.htm)
* [Modules](http://docs.google.com/modules.htm)
* [Classes](http://docs.google.com/annotated.htm)
* [Files](http://docs.google.com/files.htm)
* [Class List](http://docs.google.com/annotated.htm)
* [Class Index](http://docs.google.com/classes.htm)
* [Class Hierarchy](http://docs.google.com/hierarchy.htm)
* [Class Members](http://docs.google.com/functions.htm)
* **sf**
* [TcpListener](http://docs.google.com/classsf_1_1TcpListener.htm)

[Public Types](#_gjdgxs) | [Public Member Functions](#_30j0zll) | [Protected Types](#_1fob9te) | [Protected Member Functions](#_3znysh7) | [List of all members](http://docs.google.com/classsf_1_1TcpListener-members.htm)

sf::TcpListener Class Reference

[Network module](http://docs.google.com/group__network.htm)

[Socket](http://docs.google.com/classsf_1_1Socket.htm) that listens to new TCP connections. [More...](http://docs.google.com/classsf_1_1TcpListener.htm#details)

#include <[TcpListener.hpp](http://docs.google.com/TcpListener_8hpp_source.htm)>

Inheritance diagram for sf::TcpListener:



| Public Types | |
| --- | --- |
| enum | [Status](http://docs.google.com/classsf_1_1Socket.htm#a51bf0fd51057b98a10fbb866246176dc) {  [Done](http://docs.google.com/classsf_1_1Socket.htm#a51bf0fd51057b98a10fbb866246176dca1de3a85bc56d3ae85b3d0f3cfd04ae90),  [NotReady](http://docs.google.com/classsf_1_1Socket.htm#a51bf0fd51057b98a10fbb866246176dca8554848daae98f996e131bdeed076c09),  [Disconnected](http://docs.google.com/classsf_1_1Socket.htm#a51bf0fd51057b98a10fbb866246176dcab215141f756acdc23c67fad149710eb1),  [Error](http://docs.google.com/classsf_1_1Socket.htm#a51bf0fd51057b98a10fbb866246176dca1dc9854433a28c22e192721179a2df5d)  } |
|  | Status codes that may be returned by socket functions. [More...](http://docs.google.com/classsf_1_1Socket.htm#a51bf0fd51057b98a10fbb866246176dc) |
|  | |
| enum | { [AnyPort](http://docs.google.com/classsf_1_1Socket.htm#a5deb2c955fd347259c3a20d27b2481aaa5a3c30fd128895403afc11076f461b19) = 0 } |
|  | Some special values used by sockets. [More...](http://docs.google.com/classsf_1_1Socket.htm#a5deb2c955fd347259c3a20d27b2481aa) |
|  | |

| Public Member Functions | |
| --- | --- |
|  | [TcpListener](http://docs.google.com/classsf_1_1TcpListener.htm#a59a1db5b6f4711a3e57390da2f8d9630) () |
|  | Default constructor. |
|  | |
| unsigned short | [getLocalPort](http://docs.google.com/classsf_1_1TcpListener.htm#a3bb62b92684cd1a14b14efdaf0471440) () const |
|  | Get the port to which the socket is bound locally. |
|  | |
| [Status](http://docs.google.com/classsf_1_1Socket.htm#a51bf0fd51057b98a10fbb866246176dc) | [listen](http://docs.google.com/classsf_1_1TcpListener.htm#a409d9350d3abfea9636df8cf4a61004e) (unsigned short port) |
|  | Start listening for connections. |
|  | |
| void | [close](http://docs.google.com/classsf_1_1TcpListener.htm#a3a00a850506bd0f9f48867a0fe59556b) () |
|  | Stop listening and close the socket. |
|  | |
| [Status](http://docs.google.com/classsf_1_1Socket.htm#a51bf0fd51057b98a10fbb866246176dc) | [accept](http://docs.google.com/classsf_1_1TcpListener.htm#ae2c83ce5a64d50b68180c46bef0a7346) ([TcpSocket](http://docs.google.com/classsf_1_1TcpSocket.htm) &socket) |
|  | Accept a new connection. |
|  | |
| void | [setBlocking](http://docs.google.com/classsf_1_1Socket.htm#a165fc1423e281ea2714c70303d3a9782) (bool blocking) |
|  | Set the blocking state of the socket. |
|  | |
| bool | [isBlocking](http://docs.google.com/classsf_1_1Socket.htm#a0ec0d831b015e32eb5935fd2a9f8c67c) () const |
|  | Tell whether the socket is in blocking or non-blocking mode. |
|  | |

| Protected Types | |
| --- | --- |
| enum | [Type](http://docs.google.com/classsf_1_1Socket.htm#a5d3ff44e56e68f02816bb0fabc34adf8) {  [Tcp](http://docs.google.com/classsf_1_1Socket.htm#a5d3ff44e56e68f02816bb0fabc34adf8acc02e97e90234b957eaad4dff7f22214),  [Udp](http://docs.google.com/classsf_1_1Socket.htm#a5d3ff44e56e68f02816bb0fabc34adf8a6ebf3094830db4820191a327f3cc6ce2)  } |
|  | Types of protocols that the socket can use. [More...](http://docs.google.com/classsf_1_1Socket.htm#a5d3ff44e56e68f02816bb0fabc34adf8) |
|  | |

| Protected Member Functions | |
| --- | --- |
| SocketHandle | [getHandle](http://docs.google.com/classsf_1_1Socket.htm#ac0c63b13e61da8294bf54e888e97f9a3) () const |
|  | Return the internal handle of the socket. |
|  | |
| void | [create](http://docs.google.com/classsf_1_1Socket.htm#aafbe140f4b1921e0d19e88cf7a61dcbc) () |
|  | Create the internal representation of the socket. |
|  | |
| void | [create](http://docs.google.com/classsf_1_1Socket.htm#af1dd898f7aa3ead7ff7b2d1c20e97781) (SocketHandle handle) |
|  | Create the internal representation of the socket from a socket handle. |
|  | |

## Detailed Description

[Socket](http://docs.google.com/classsf_1_1Socket.htm) that listens to new TCP connections.

A listener socket is a special type of socket that listens to a given port and waits for connections on that port.

This is all it can do.

When a new connection is received, you must call accept and the listener returns a new instance of [sf::TcpSocket](http://docs.google.com/classsf_1_1TcpSocket.htm) that is properly initialized and can be used to communicate with the new client.

[Listener](http://docs.google.com/classsf_1_1Listener.htm) sockets are specific to the TCP protocol, UDP sockets are connectionless and can therefore communicate directly. As a consequence, a listener socket will always return the new connections as [sf::TcpSocket](http://docs.google.com/classsf_1_1TcpSocket.htm) instances.

A listener is automatically closed on destruction, like all other types of socket. However if you want to stop listening before the socket is destroyed, you can call its [close()](http://docs.google.com/classsf_1_1TcpListener.htm#a3a00a850506bd0f9f48867a0fe59556b) function.

Usage example:

// Create a listener socket and make it wait for new

// connections on port 55001

[sf::TcpListener](http://docs.google.com/classsf_1_1TcpListener.htm) listener;

listener.[listen](http://docs.google.com/classsf_1_1TcpListener.htm#a409d9350d3abfea9636df8cf4a61004e)(55001);

// Endless loop that waits for new connections

while (running)

{

[sf::TcpSocket](http://docs.google.com/classsf_1_1TcpSocket.htm) client;

if (listener.[accept](http://docs.google.com/classsf_1_1TcpListener.htm#ae2c83ce5a64d50b68180c46bef0a7346)(client) == [sf::Socket::Done](http://docs.google.com/classsf_1_1Socket.htm#a51bf0fd51057b98a10fbb866246176dca1de3a85bc56d3ae85b3d0f3cfd04ae90))

{

// A new client just connected!

std::cout << "New connection received from " << client.[getRemoteAddress](http://docs.google.com/classsf_1_1TcpSocket.htm#a7904ca6ab9e018021e305a3aeb7a1b9a)() << std::endl;

doSomethingWith(client);

}

}

See Also[sf::TcpSocket](http://docs.google.com/classsf_1_1TcpSocket.htm), [sf::Socket](http://docs.google.com/classsf_1_1Socket.htm)

Definition at line [43](http://docs.google.com/TcpListener_8hpp_source.htm#l00043) of file [TcpListener.hpp](http://docs.google.com/TcpListener_8hpp_source.htm).

## Member Enumeration Documentation

| | anonymous enum | | --- | | inherited |
| --- | --- | --- |

Some special values used by sockets.

**Enumerator:**

| *AnyPort* | Special value that tells the system to pick any available port. |
| --- | --- |

Definition at line [65](http://docs.google.com/Socket_8hpp_source.htm#l00065) of file [Socket.hpp](http://docs.google.com/Socket_8hpp_source.htm).

| | enum [sf::Socket::Status](http://docs.google.com/classsf_1_1Socket.htm#a51bf0fd51057b98a10fbb866246176dc) | | --- | | inherited |
| --- | --- | --- |

Status codes that may be returned by socket functions.

**Enumerator:**

| *Done* | The socket has sent / received the data. |
| --- | --- |
| *NotReady* | The socket is not ready to send / receive data yet. |
| *Disconnected* | The TCP socket has been disconnected. |
| *Error* | An unexpected error happened. |

Definition at line [53](http://docs.google.com/Socket_8hpp_source.htm#l00053) of file [Socket.hpp](http://docs.google.com/Socket_8hpp_source.htm).

| | enum [sf::Socket::Type](http://docs.google.com/classsf_1_1Socket.htm#a5d3ff44e56e68f02816bb0fabc34adf8) | | --- | | protectedinherited |
| --- | --- | --- |

Types of protocols that the socket can use.

**Enumerator:**

| *Tcp* | TCP protocol. |
| --- | --- |
| *Udp* | UDP protocol. |

Definition at line [113](http://docs.google.com/Socket_8hpp_source.htm#l00113) of file [Socket.hpp](http://docs.google.com/Socket_8hpp_source.htm).

## Constructor & Destructor Documentation

| sf::TcpListener::TcpListener | ( |  | ) |  |
| --- | --- | --- | --- | --- |

Default constructor.

## Member Function Documentation

| [Status](http://docs.google.com/classsf_1_1Socket.htm#a51bf0fd51057b98a10fbb866246176dc) sf::TcpListener::accept | ( | [TcpSocket](http://docs.google.com/classsf_1_1TcpSocket.htm) & | *socket* | ) |  |
| --- | --- | --- | --- | --- | --- |

Accept a new connection.

If the socket is in blocking mode, this function will not return until a connection is actually received.

Parameters

| socket | [Socket](http://docs.google.com/classsf_1_1Socket.htm) that will hold the new connection |
| --- | --- |

ReturnsStatus code See Also[listen](http://docs.google.com/classsf_1_1TcpListener.htm#a409d9350d3abfea9636df8cf4a61004e)

| void sf::TcpListener::close | ( |  | ) |  |
| --- | --- | --- | --- | --- |

Stop listening and close the socket.

This function gracefully stops the listener. If the socket is not listening, this function has no effect.

See Also[listen](http://docs.google.com/classsf_1_1TcpListener.htm#a409d9350d3abfea9636df8cf4a61004e)

| | void sf::Socket::create | ( |  | ) |  | | --- | --- | --- | --- | --- | | protectedinherited |
| --- | --- | --- | --- | --- | --- | --- |

Create the internal representation of the socket.

This function can only be accessed by derived classes.

| | void sf::Socket::create | ( | SocketHandle | *handle* | ) |  | | --- | --- | --- | --- | --- | --- | | protectedinherited |
| --- | --- | --- | --- | --- | --- | --- | --- |

Create the internal representation of the socket from a socket handle.

This function can only be accessed by derived classes.

Parameters

| handle | OS-specific handle of the socket to wrap |
| --- | --- |

| | SocketHandle sf::Socket::getHandle | ( |  | ) | const | | --- | --- | --- | --- | --- | | protectedinherited |
| --- | --- | --- | --- | --- | --- | --- |

Return the internal handle of the socket.

The returned handle may be invalid if the socket was not created yet (or already destroyed). This function can only be accessed by derived classes.

ReturnsThe internal (OS-specific) handle of the socket

| unsigned short sf::TcpListener::getLocalPort | ( |  | ) | const |
| --- | --- | --- | --- | --- |

Get the port to which the socket is bound locally.

If the socket is not listening to a port, this function returns 0.

ReturnsPort to which the socket is bound See Also[listen](http://docs.google.com/classsf_1_1TcpListener.htm#a409d9350d3abfea9636df8cf4a61004e)

| | bool sf::Socket::isBlocking | ( |  | ) | const | | --- | --- | --- | --- | --- | | inherited |
| --- | --- | --- | --- | --- | --- | --- |

Tell whether the socket is in blocking or non-blocking mode.

ReturnsTrue if the socket is blocking, false otherwise See Also[setBlocking](http://docs.google.com/classsf_1_1Socket.htm#a165fc1423e281ea2714c70303d3a9782)

| [Status](http://docs.google.com/classsf_1_1Socket.htm#a51bf0fd51057b98a10fbb866246176dc) sf::TcpListener::listen | ( | unsigned short | *port* | ) |  |
| --- | --- | --- | --- | --- | --- |

Start listening for connections.

This functions makes the socket listen to the specified port, waiting for new connections. If the socket was previously listening to another port, it will be stopped first and bound to the new port.

Parameters

| port | Port to listen for new connections |
| --- | --- |

ReturnsStatus code See Also[accept](http://docs.google.com/classsf_1_1TcpListener.htm#ae2c83ce5a64d50b68180c46bef0a7346), [close](http://docs.google.com/classsf_1_1TcpListener.htm#a3a00a850506bd0f9f48867a0fe59556b)

| | void sf::Socket::setBlocking | ( | bool | *blocking* | ) |  | | --- | --- | --- | --- | --- | --- | | inherited |
| --- | --- | --- | --- | --- | --- | --- | --- |

Set the blocking state of the socket.

In blocking mode, calls will not return until they have completed their task. For example, a call to Receive in blocking mode won't return until some data was actually received. In non-blocking mode, calls will always return immediately, using the return code to signal whether there was data available or not. By default, all sockets are blocking.

Parameters

| blocking | True to set the socket as blocking, false for non-blocking |
| --- | --- |

See Also[isBlocking](http://docs.google.com/classsf_1_1Socket.htm#a0ec0d831b015e32eb5935fd2a9f8c67c)

The documentation for this class was generated from the following file:

* [TcpListener.hpp](http://docs.google.com/TcpListener_8hpp_source.htm)

Copyright � Laurent Gomila  ::  Documentation generated by [doxygen](http://www.doxygen.org/)  ::