<http://blog.csdn.net/iloveqt5/article/details/19471587>

[QTcpServer](http://qt-project.org/doc/qt-5/qtcpserver.html)用函数

|  |  |
| --- | --- |
| bool | [listen](http://qt-project.org/doc/qt-5/qtcpserver.html#listen)(const QHostAddress & address = QHostAddress::Any, quint16 port = 0) |

监听某个端口，如果有新连接进来就发出[newConnection](http://qt-project.org/doc/qt-5/qtcpserver.html#newConnection)()信号，连接过程中出错的话就发出[acceptError](http://qt-project.org/doc/qt-5/qtcpserver.html#acceptError)(QAbstractSocket::SocketError socketError)信号。

在接受连接的过程中，应当是有一个队列，等待的连接都将放入连接队列中，名词PendingConnections中的Pending意为即将发生的，在等待中的。

当队列中有等待的连接时，可用函数

|  |  |
| --- | --- |
| virtual QTcpSocket \* | [nextPendingConnection](http://qt-project.org/doc/qt-5/qtcpserver.html#nextPendingConnection)() |

获得这个连接，并为这个连接分配一部"电话"（即函数返回的QTcpSocket），通信就建立了。然后可通过这个返回的QTcpSocket与客户端通信。值得注意的是，这个返回的socket不可用于另一个线程。

这是一种方式，另一种方式可继承这个类，重写它的Protected Function：

virtual void [incomingConnection](http://qt-project.org/doc/qt-5/qtcpserver.html#incomingConnection)(qintptr socketDescriptor)

这个函数在有连接时自动执行。

socketDescriptor为套接字描述符，通俗理解为，系统为这个连接进来的套接字自动分配了个ID，用于区分不同套接字，是个整数值。将这个ID传给套接字就行了。网上说它和文件的句柄类似。

————————————————————————————————————————————————————————————

下面看下这个[QTcpServer](http://qt-project.org/doc/qt-5/qtcpserver.html)

公有函数：

**[cpp]** [view plain](http://blog.csdn.net/iloveqt5/article/details/19471587) [copy](http://blog.csdn.net/iloveqt5/article/details/19471587)

1. QTcpServer(QObject \* parent = 0)
2. **virtual** ~QTcpServer()
4. **void**    setMaxPendingConnections(**int** numConnections)//设置等待连接队列中连接的最大数量，默认30
5. **void**    setProxy(**const** QNetworkProxy & networkProxy)//为它设置代理
6. **bool**    setSocketDescriptor(qintptr socketDescriptor)//手动设置套接字描述符，取代系统自动分配
7. **bool**    listen(**const** QHostAddress & address = QHostAddress::Any, quint16 port = 0)//监听某端口
9. **virtual** QTcpSocket \*    nextPendingConnection()//获得队列中的下一个连接
10. **void**    pauseAccepting()//停止接受新连接
11. **void**    resumeAccepting()//重启接受新连接
12. **void**    close()
13. **bool**    waitForNewConnection(**int** msec = 0, **bool** \* timedOut = 0)//阻塞方式接受新连接，msec为设置超时时间，单位毫秒
15. QAbstractSocket::SocketError    serverError() **const**
16. QString     errorString() **const**//产生错误的字符串
17. **int**     maxPendingConnections() **const**
18. QNetworkProxy   proxy() **const**
19. qintptr socketDescriptor() **const**
20. QHostAddress    serverAddress() **const**
21. quint16     serverPort() **const**
22. **bool**        isListening() **const**
23. **virtual** **bool**    hasPendingConnections() **const**

Signals

|  |  |
| --- | --- |
| void | [acceptError](http://qt-project.org/doc/qt-5/qtcpserver.html#acceptError)(QAbstractSocket::SocketError socketError) |
| void | [newConnection](http://qt-project.org/doc/qt-5/qtcpserver.html#newConnection)() |

Protected Functions

|  |  |
| --- | --- |
| void | [addPendingConnection](http://qt-project.org/doc/qt-5/qtcpserver.html#addPendingConnection)(QTcpSocket \* socket) |
| virtual void | [incomingConnection](http://qt-project.org/doc/qt-5/qtcpserver.html#incomingConnection)(qintptr socketDescriptor) |