**版本修订表**

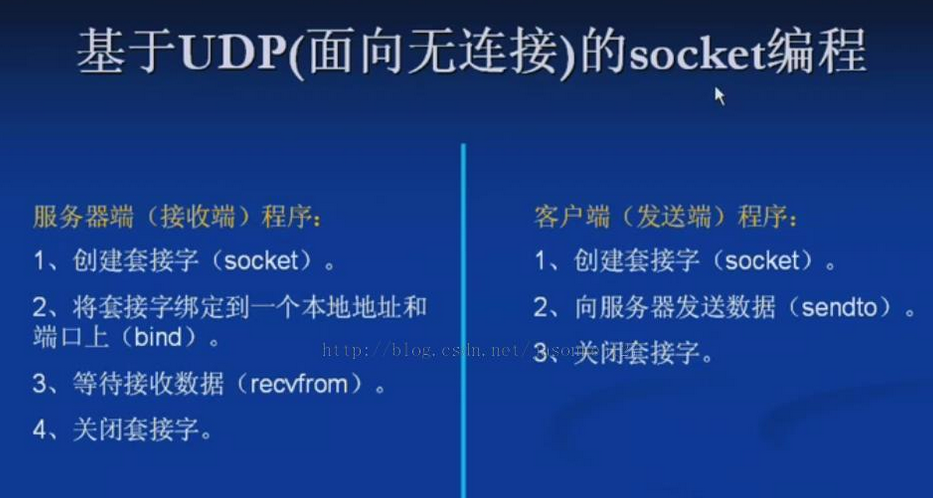
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | 状态 | 日期 | 修改人 | 说明 |
| V1.0 | 新建 | 2016.10.29 | 易嘉祯 | 编写一期设计方案 |

# 一 LQNetworkLib背景介绍

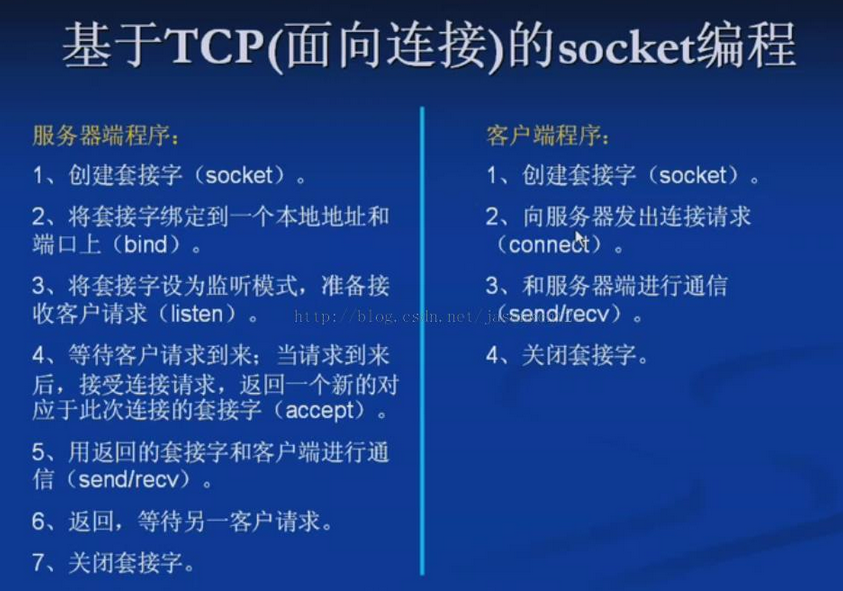
封装该lib库主要用于LuckyQQ项目内部通讯功能的使用。内部需实现TCP，UDP连接的两种方式，可以并发处理多路请求，可以使用阻塞和非阻塞两种模式。

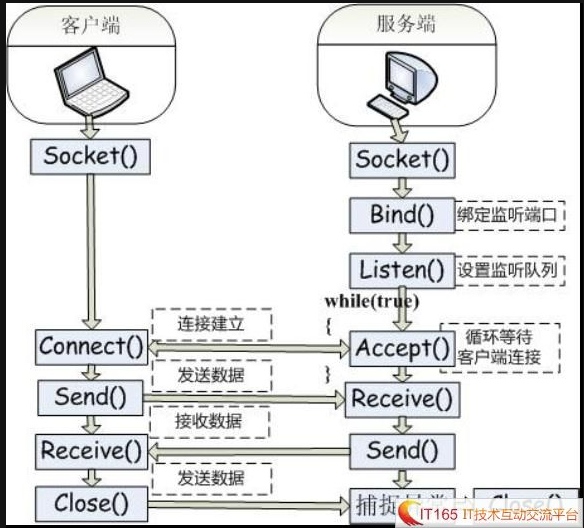
# 二 LQNetworkLib通讯方式

## 1 基于UDP(面向无连接)的socket编程



## 2 基于TCP(面向连接)的socket编程





# 三 LQNetworkLib类与接口设计

基类与派生关系图：

Address

sockaddr\_in

CommonSocket

UDP

TCP

## Address类的设计

Class Address: protected sockaddr\_in

{

Public:

Address(); // 默认端口0, 默认ip地址127.0.0.1

Address(const Port& inPort); // 只传端口，则默认ip地址为127.0.0.1

Address(const IP& inIP, const Port& inPort);

Address(struct sockaddr\_in inSockAddr);

Address(const Address& inAddress);

IP getIP(void);

Void setIP(const IP& inIP);

Port getPort(void);

Void setPort(const Port& inPort);

Friend ostream& operator << (ostream&out, Address& inAddress); // 显示：ip:port

}

## CommonSocket类的设计

Class CommonSocket

{

Public:

CommonSocket(void);

CommonSocket(int inSocketType);

~CommonSocket(void);

Void open(void); // 创建socket

Void close(void); // 关闭socket

void bind\_to\_port(const Port& inPort); // 将socket绑定到端口上

Protected:

SocketID m\_socketID;

Int m\_socketType;

Bool m\_bOpened;

Bool m\_bBinded;

}

## UDP类的设计

Class UDP:public CommonSocket

{

Public:

UDP(void);

UDP(int inIOType); // 设置IO方式：（1.阻塞 2.非阻塞）

UDP(const UDP& inUDP);

Int send(const IP& inIP, const Port& port, const char\* inContent, const size\_t& inSize);

Int send(Address inAddress, const char\* inContent, const size\_t& inSize);

Int receive(IP\* outIP, Port\* outPort, char\* outContent, size\_t\* outSize);

Int receive(Address\* outAddress, char\* outContent, size\_t\* outSize);

}

## TCP类的设计

Class TCP: public CommonSocket

{

Public:

TCP(void);

TCP(int inIOType); // 设置IO方式：（1.阻塞 2.非阻塞）

TCP(const TCP& inTCP);

IP getIP(void);

Port getPort(void);

Address getAddress(void);

int listen\_on\_port(const Port& inPort, const unsigned int& listeners);

int connect\_to(Address inAddress);

int accept\_client(TCP\* outTCP);

int send(const char\* inContent, const size\_t& inSize);

int receive(char\* outContent, size\_t\* outSize);

Private:

Address m\_address;

}