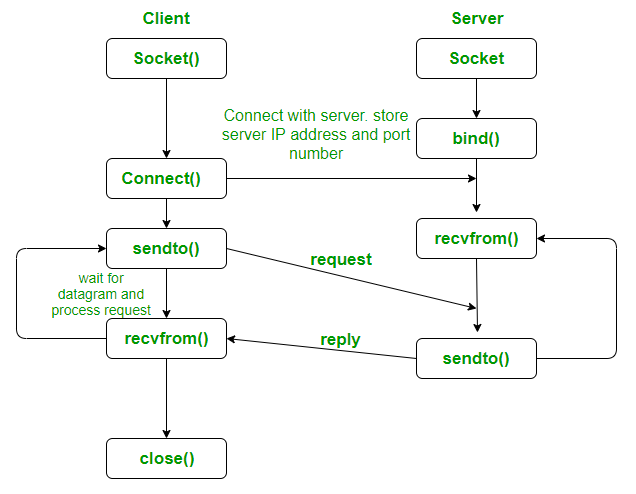
**UDP Client Server using connect | C implementation**

UDP is a connection less protocol. There is no connection is established between client and server.

In UDP, the client does not form a connection with the server like in TCP and instead, It just sends a datagram. Similarly, the server need not to accept a connection and just waits for datagrams to arrive. We can call a function called **connect()** in UDP but it does not result anything like it does in TCP. There is no 3 way handshake. It just checks for any immediate errors and store the peer’s IP address and port number. connect() is storing peers address so no need to pass **server address** and **server address length** arguments in **sendto()**.  


**int connect(int sockfd, const struct sockaddr \*servaddr,**

**socklen\_t addrlen);**

returns : 0 if OK -1 on error

**arguments :**

**sockfd** : File descriptor of socket to be connected.

**struct sockaddr \*servaddr** : server address structure.

**addrlen** : length of server address structure.

// server program for udp connection

UDP Server code :

#include <stdio.h>

#include <strings.h>

#include <sys/types.h>

#include <arpa/inet.h>

#include <sys/socket.h>

#include<netinet/in.h>

#define PORT 5000

#define MAXLINE 1000

// Driver code

int main()

{

char buffer[100];

char \*message = "Hello Client";

int listenfd, len;

struct sockaddr\_in servaddr, cliaddr;

bzero(&servaddr, sizeof(servaddr));

// Create a UDP Socket

listenfd = socket(AF\_INET, SOCK\_DGRAM, 0);

servaddr.sin\_addr.s\_addr = htonl(INADDR\_ANY);

servaddr.sin\_port = htons(PORT);

servaddr.sin\_family = AF\_INET;

// bind server address to socket descriptor

bind(listenfd, (struct sockaddr\*)&servaddr, sizeof(servaddr));

//receive the datagram

len = sizeof(cliaddr);

int n = recvfrom(listenfd, buffer, sizeof(buffer), 0, (struct sockaddr\*)&cliaddr,&len); //receive message from server

buffer[n] = '\0';

puts(buffer);

// send the response

sendto(listenfd, message, MAXLINE, 0,

(struct sockaddr\*)&cliaddr, sizeof(cliaddr));

}

UDP Client code :

// udp client driver program

#include <stdio.h>

#include <strings.h>

#include <sys/types.h>

#include <arpa/inet.h>

#include <sys/socket.h>

#include<netinet/in.h>

#include<unistd.h>

#include<stdlib.h>

#define PORT 5000

#define MAXLINE 1000

// Driver code

int main()

{

char buffer[100];

char \*message = "Hello Server";

int sockfd, n;

struct sockaddr\_in servaddr;

// clear servaddr

bzero(&servaddr, sizeof(servaddr));

servaddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

servaddr.sin\_port = htons(PORT);

servaddr.sin\_family = AF\_INET;

// create datagram socket

sockfd = socket(AF\_INET, SOCK\_DGRAM, 0);

// connect to server

if(connect(sockfd, (struct sockaddr \*)&servaddr, sizeof(servaddr)) < 0)

{

printf("\n Error : Connect Failed \n");

exit(0);

}

// request to send datagram

// no need to specify server address in sendto

// connect stores the peers IP and port

sendto(sockfd, message, MAXLINE, 0, (struct sockaddr\*)NULL, sizeof(servaddr));

// waiting for response

recvfrom(sockfd, buffer, sizeof(buffer), 0, (struct sockaddr\*)NULL, NULL);

puts(buffer);

// close the descriptor

close(sockfd);

}