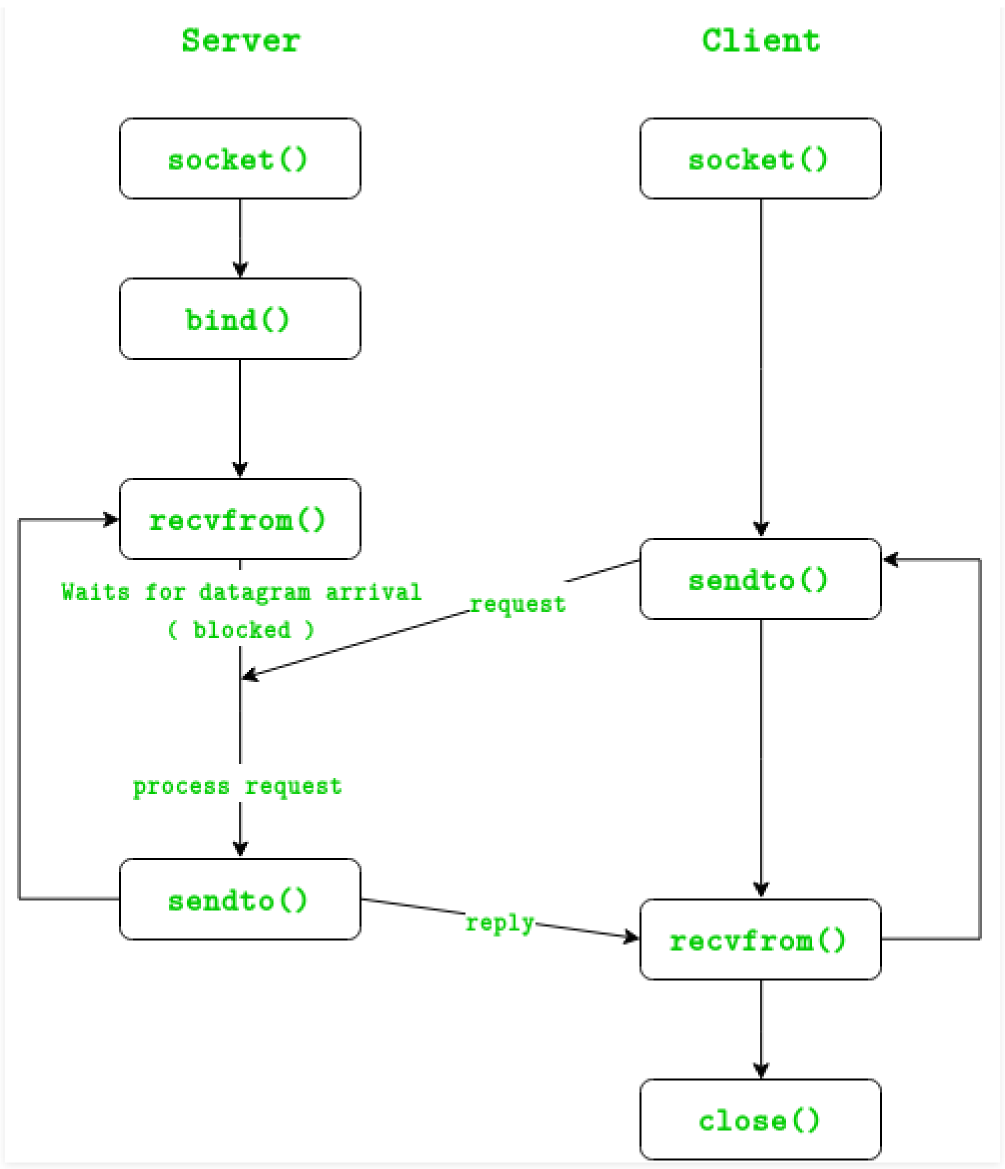
Few important points

I. The client does not form a connection with the server like in TCP and instead just sends a datagram.

II. Similarly, the server need not accept a connection and just waits for datagrams to arrive



UDP Server:

1. Create UDP socket.
2. Bind the socket to server address.
3. Wait until datagram packet arrives from client.
4. Process the datagram packet and send a reply to client.
5. Go back to Step 3.

>> Sample Server side dummy code in UDP has been written below

```
#define PORT     8080
#define MAXLINE 1024

int main() {
   int sockfd;
   char buffer[MAXLINE];
   char *hello = "Hello from server";
   struct sockaddr_in servaddr, cliaddr;

   // TODO: 1 Creating socket file descriptor using SOCK_DGRAM
   if ( (sockfd = socket()) < 0 ) {
        // print error
        // Clean up the module
   }

   // TODO: 2 Clean up the sockaddr structures
   // TODO: 3 Filling server information in servaddr
   // TODO: 4 Bind the socket with the server address
   if ( bind() < 0 ) {
         // print error
        // Clean up the module
   }

   // TODO: 5 Make a blank entry for cliaddr
   // TODO: 6 Directly use recvfrom() api to receive data from client
   n = recvfrom(buffer);

   // print the buffer in case of string
   buffer[n] = '\0';
   printf("Client : %s\n", buffer);

   // TODO: 7 send reply to the client using sendto()
   sendto();

   return 0;
}
```

UDP Client:

1. Create UDP socket.
2. Send message to server.
3. Wait until response from server is received.
4. Process reply and go back to step 2, if necessary.
5. Close socket descriptor and exit.

>> Sample Client side dummy code in UDP has been written below

```
#define PORT     8080
#define MAXLINE 1024

int main() {
   int sockfd;
   char buffer[MAXLINE];
   char *hello = "Hello from client";
   struct sockaddr_in     servaddr;

   // TODO: 1 Creating socket file descriptor using SOCK_DGRAM
   if ( (sockfd = socket()) < 0 ) {
        // print error
        // Clean up the module
   }

   // TODO: 2 Clean up the sockaddr structure servaddr
   // TODO: 3 Filling server information in servaddr
   // TODO: 4 send reply to the client using sendto()
   // TODO: 5 Directly use recvfrom() api to receive data from server
   n = recvfrom(buffer);

   // print the buffer in case of string
   buffer[n] = '\0';
   printf("Server : %s\n", buffer);

   close(sockfd);
   return 0;
}
```