**IMPLEMENT - DNS USING UDP SOCKETS**

**AIM:**

To implement DNS using UDP sockets in java.

**THEORY:**

Short for **D**omain **N**ame **S**ystem (or **S**ervice or **S**erver), an [Internet](http://www.webopedia.com/TERM/I/Internet.html) service that translates [domain names](http://www.webopedia.com/TERM/D/domain_name.html) into IP addresses. Because domain names are alphabetic, they're easier to remember. The Internet however, is really based on [IP addresses](http://www.webopedia.com/TERM/I/IP_address.html). Every time you use a domain name, therefore, a DNS service must translate the name into the corresponding IP address. For example, the domain name www.example.com might translate to 198.105.232.4.

The Domain Name System distributes the responsibility of assigning domain names and mapping those names to IP addresses by designating [authoritative name servers](http://en.wikipedia.org/wiki/Authoritative_name_server) for each domain. Authoritative name servers are assigned to be responsible for their supported domains, and may delegate authority over subdomains to other name servers. This mechanism provides distributed and fault tolerant service and was designed to avoid the need for a single central database.The DNS system is, in fact, its own [network](http://www.webopedia.com/TERM/N/network.html). If one DNS server doesn't know how to translate a particular domain name, it asks another one, and so on, until the correct IP address is returned.

**ALGORITHM:**

1. Start the program

2. Enter the system name (dn) after the connection with the server is established.

3. Call the subroutine resolver and pass the system name to the resolver.

4. Open the host files in the DNS server.

5. Check the system name with the name stored in the host file until the end of the file.

6. If the name matches, then fetch the corresponding IP address and display the IP address. Go to step8.

7. If the match is not found, then display the message as system is not logged on.

8. Stop the process.

**PROGRAM**

DNS CLIENT

import java.io.\*;

import java.net.\*;

importjava.util.\*;

classdclient

{

public static void main(String args[])

{

try

{

DatagramSocket client=new DatagramSocket();

InetAddressaddr=InetAddress.getByName("127.0.0.1");

byte[]sendbyte=new byte[1024];

byte[]receivebyte=new byte[1024];

BufferedReader in=new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter DOMAIN NAME OR IP address");

String str=in.readLine();

sendbyte=str.getBytes();

DatagramPacket sender=new DatagramPacket(sendbyte,sendbyte.length,addr,1309);

client.send(sender);

DatagramPacket receiver= new DatagramPacket(receivebyte,receivebyte.length);

client.receive(receiver);

String s=new String(receiver.getData());

System.out.println("IP adddress or DOMAIN NAME :"+s.trim());

client.close();

}

catch(Exception e)

{

System.out.println(e);

}

}

}

DNS SERVER

import java.io.\*;

import java.net.\*;

import java.util.\*;

class dserver

{

public static void main(String args[])

{

try

{

DatagramSocket server=new DatagramSocket(1309);

while(true)

{

byte[]sendbyte=new byte[1024];

byte[]receivebyte=new byte[1024];

DatagramPacket receiver=new DatagramPacket(receivebyte,receivebyte.length);

server.receive(receiver);

String str=new String(receiver.getData());

String s=str.trim();

//system.out.println(s);

InetAddressaddr=receiver.getAddress();

int port=receiver.getPort();

String ip[]={"165.165.80.80","165.165.79.1"};

String name[]={"www.aptitude source.com","www.Sharifguys.com"};

for(int i=0;i<ip.length;i++)

{

if(s.equals(ip[i]))

{

sendbyte=name[i].getBytes();

DatagramPacket sender=new DatagramPacket(sendbyte,sendbyte.length,addr, port);

server.send(sender);

break;

}

else if(s.equals(name[i]))

{

sendbyte=ip[i].getBytes();

DatagramPacket sender=new DatagramPacket(sendbyte,sendbyte.length,addr, port);

server.send(sender);

break;

}

}

break;

}

}

catch(Exception e)

{

System.out.println(e);

}

}

}

**OUTPUT:**

**Java Server**

165.165.80.80

165.165.79.1

**Java Client**

Enter domain name or IP address

165.165.80.80

Ip address or domain name: www.aptitude source.com

165.165.79.1

Ip address or domain name: www.Sharifguys.com

**DNS**

**AIM :**

To write a C program to implement DNS using UDP sockets.

# PROGRAM :

**SERVER**

#include<sys/socket.h> #include<unistd.h> #include<stdio.h> #include<string.h> #include<netdb.h> #include<stdlib.h> #include<arpa/inet.h> #define MAX 100

#define PORT 9999 #define SA struct sockaddr int main()

{

int sockfd;

struct sockaddr\_in servaddr; struct in\_addr address; struct hostent \*hp;

sockfd=socket(AF\_INET,SOCK\_DGRAM,0); if(sockfd==-1)

{

printf("Socket creation failed\n"); exit(0);

}

else

printf("Socket successfully created\n"); bzero(&servaddr,sizeof(servaddr)); servaddr.sin\_family=AF\_INET; servaddr.sin\_addr.s\_addr=htonl(INADDR\_ANY); servaddr.sin\_port=htons(PORT);

if((bind(sockfd,(SA \*)&servaddr,sizeof(servaddr)))!=0)

{

printf("Error in Binding\n"); exit(0);

}

else

printf("Successfully Binded \n"); char buff[MAX];

int n,clen;

struct sockaddr\_in cli; clen=sizeof(cli); while(1)

{

bzero(buff,MAX); recvfrom(sockfd,buff,sizeof(buff),0,(SA \*)&cli,&clen); if(strncmp("exit",buff,4)==0)

{

printf("\nServer exit\n"); break;

}

printf("\n->URL given by client %s\n",buff); int bufferlength = strlen(buff); buff[bufferlength - 1] = '\0';

hp = gethostbyname(buff); if(hp)

{

bcopy(\*hp->h\_addr\_list, (char\*) &address , sizeof(address)); sendto(sockfd,inet\_ntoa(address),sizeof(buff),0,(SA \*)&cli,clen);

}

else

{

// ECHO

sendto(sockfd,buff,sizeof(buff),0,(SA \*)&cli,clen);

}

if(strncmp("exit",buff,4)==0)

{

printf("\n->Server exit\n"); break;

}

}

close(sockfd);

}

# CLIENT

#include<sys/socket.h> #include<netdb.h> #include<string.h> #include<stdlib.h> #include <sys/types.h> #include<stdio.h> #include <arpa/inet.h> #define MAX 100

#define PORT 9999 #define SA struct sockaddr int main()

{

char buff[MAX]; int sockfd,len,n;

struct sockaddr\_in servaddr; sockfd=socket(AF\_INET,SOCK\_DGRAM,0); if(sockfd==-1)

{

printf("Socket creation failed\n"); exit(0);

}

else

printf("Socket successfully created\n"); bzero(&servaddr,sizeof(len)); servaddr.sin\_family=AF\_INET; servaddr.sin\_addr.s\_addr=inet\_addr("127.0.0.1"); servaddr.sin\_port=htons(PORT); len=sizeof(servaddr);

while(1)

{

printf("->Enter the URL: "); fgets(buff,MAX,stdin); if(strncmp("exit",buff,4)==0)

{

printf("\n->Client exit\n"); sendto(sockfd,buff,sizeof(buff),0,(SA \*)&servaddr,len); break;

}

sendto(sockfd,buff,sizeof(buff),0,(SA \*)&servaddr,len); bzero(buff,sizeof(buff)); recvfrom(sockfd,buff,sizeof(buff),0,(SA \*)&servaddr,&len);

printf("\n->IP Address : %s\n",buff);

}

close(sockfd);

}

# OUTPUT :

SERVER :



CLIENT :



# RESULT :

A C program for implementing DNS using UDP sockets is written and executed successfully.