**Прост сървър**

```
#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
int main()
{
        int server_sockfd, client_sockfd;
        int server_len, client_len;
        struct sockaddr_in server_address;
        struct sockaddr_in client_address;
        /* Create an unnamed socket for the server. */
        server_sockfd = socket(AF_INET, SOCK_STREAM, 0);
        /* Name the socket. */
        server_address.sin_family = AF_INET;
        /* Point interface (INADDR_ANY if any) */
     server_address.sin_addr.s_addr = inet_addr("127.0.0.1");
        server_address.sin_port = 9734;
        server_len = sizeof(server_address);
     bind(server_sockfd, (struct sockaddr *)&server_address,
        server_len);
        /* Create a connection queue and wait for clients.
        */
        listen(server_sockfd, 5);
        while(1) {
                char ch;
                printf("server waiting\n");

                /* Accept a connection. */
                client_len = sizeof(client_address);
                client_sockfd = accept(server_sockfd,
              (struct sockaddr *)&client_address, &client_len);

                /* We can now read/write to client on
                client_sockfd. */
                while( read(client_sockfd, &ch, 1) !=0)
                printf (“server receives =%c\n”,ch);
                printf(“server closes\n”);
                close(client_sockfd);
        }
}
```

**Прост клиент**

```
#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
int main()
{
        int sockfd;
        int client_len;
        int i;
        struct sockaddr_in client_address;
        int result;
        char ch = 'A';
        /* Create a socket for the client. */
        sockfd = socket(AF_INET, SOCK_STREAM, 0);
        /* Name the socket, as agreed with the server. */
        client_address.sin_family = AF_INET;
        client_address.sin_addr.s_addr =inet_addr("127.0.0.1");
        client_address.sin_port = 9734;
        client_len = sizeof(client_address);
        /* Now connect our socket to the server's socket.
        */
result = connect(sockfd, (struct sockaddr*) &client_address,
client_len);
        if(result == -1) {
        perror("oops: clienta");
        exit(1);
        }
        /* We can now read/write via sockfd. */
        for (i=0; i<=9;i++) {
        write(sockfd, &ch,1);
        printf("client sends= %c\n", ch);
        ch++;
        }
        close(sockfd);
        exit(0);
}
```

**Подобрен клиент**

```
//Следващият код показва, как сърверът може да се
//специфицира с име:
#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <netdb.h>

int main(int argc, char *argv[] )
{
        int sockfd;
        int client_len;
        int i;
        char* host;
        struct sockaddr_in client_address;
        struct hostent *hostinfo;
        int result;
        char ch = 'A';
        if(argc == 1)
                host = "localhost";
        else
                host = argv[1];

        /* Find the host address and report an error if
        none is found. */
        hostinfo = gethostbyname(host);
        if(!hostinfo) {
                fprintf(stderr, "no host: %s\n", host);
        exit(1); }
        /* Create a socket for the client. */
        sockfd = socket(AF_INET, SOCK_STREAM, 0);
        /* Name the socket, as agreed with the server. */
        client_address.sin_family = AF_INET;
client_address.sin_addr = *(struct in_addr *)*hostinfo ->
        h_addr_list;
        client_address.sin_port =htons(9734);
        client_len = sizeof(client_address);
        /* Now connect our socket to the server's socket
        and etc...*/
```

**Пример – инфомация за сървера**

```
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <netdb.h>
#include <stdio.h>
int main(int argc, char *argv[])
{
        char *host, **names, **addrs;
        struct hostent *hostinfo;
        /* Set the host in question to the argument
        supplied with the getname call,
        or default to the user's machine. */
        if(argc == 1) {
                char myname[256];
                gethostname(myname, 255);
        host = myname;
        }else host = argv[1];
     /* Make the call to gethostbyname and report an error if
        no information is found. */
        hostinfo = gethostbyname(host);
        if(!hostinfo) {
                fprintf(stderr, "cannot get info for host:
                %s\n", host);
                exit(1);
        }
     /* Display the hostname and any aliases it may have. */
        printf("results for host %s:\n", host);
        printf("Name: %s\n", hostinfo -> h_name);
        printf("Aliases:");
        names = hostinfo -> h_aliases;
        while(*names) {
                printf(" %s", *names);
                names++;
        }printf("\n");
     /* Warn and exit if the host in question isn't an IP host. */
        if(hostinfo -> h_addrtype != AF_INET) {
                fprintf(stderr, "not an IP host!\n");
                exit(1);
        }
     /* Otherwise, display the IP address(es). */
        addrs = hostinfo -> h_addr_list;
        while(*addrs)
        {
            printf(" %s", inet_ntoa(*(struct in_addr *)*addrs));
                addrs++;
        }printf("\n");
        exit(0);
}
```

**Сървър за много клиенти**

```
//За всеки клиент се създава отделен процес. Пример:
#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <netinet/in.h>
#include <signal.h>
#include <unistd.h>

int main()
{
        int server_sockfd, client_sockfd;
        int server_len, client_len;
        int server_n = 0;
        struct sockaddr_in server_address;
        struct sockaddr_in client_address;
        server_sockfd = socket(AF_INET, SOCK_STREAM, 0);
        server_address.sin_family = AF_INET;
        server_address.sin_addr.s_addr =
        htonl(INADDR_ANY);
        server_address.sin_port = 9734;
        server_len = sizeof(server_address);
     bind(server_sockfd, (struct sockaddr *)&server_address,
        server_len);
      /* Create a connection queue, ignore child exit
        details and wait for clients. */
        listen(server_sockfd, 5);
        signal(SIGCHLD, SIG_IGN);
        while(1) {
                char ch;
                printf("server waiting\n");
     /* Accept connection. */
        client_len = sizeof(client_address);
        client_sockfd = accept(server_sockfd,
        (struct sockaddr *)&client_address, &client_len);
     /* Fork to create a process for this client*/
        server_n++;
        if(fork() == 0) {
     /* If we're the child, we can now read/write to the client on
client_sockfd. The five second delay is a demonstration. */
                while (read(client_sockfd, &ch, 1)!=0) {
                printf("server %d receives=%c\n",server_n,ch);
                        sleep(5);
                }
        printf ("server closes\n");
        close(client_sockfd);
        exit(0);
        }
        /* Otherwise, we must be the parent and our work
        for this client is finished. */
        else {
                close(client_sockfd);
                }
        }
}
```

**Пример Select**

```
/* ** select.c -- a select() demo*/
#include <stdio.h>
#include <sys/time.h>
#include <sys/types.h>
#include <unistd.h>
#define STDIN 0 // standard input descriptor
int main()
{
        struct timeval tv;
        fd_set readfds;
        tv.tv_sec = 2;
        v.tv_usec = 500000;
        FD_ZERO(&readfds);
        FD_SET(STDIN, &readfds);
        // don't care about writefds and exceptfds:
        select(STDIN+1, &readfds, NULL, NULL, &tv);
        if (FD_ISSET(STDIN, &readfds))
                printf("A key was pressed!\n");
        else
        printf("Timed out.\n");
        return 0;
        }
```

**Схема на сървер, обслужващ много клиенти, от които той само чете със Select**

```
/*множество дескриптори setd, setw;
        Създаване на socket на услугата – sockserv; */
        setw = {sockserv };
        while (1) {
                setd = setw;
                select (..,setd,NULL,NULL,NULL);
                for i in setd {
                if (i == sockserv) {
                        sockconn = accept(..);
                        setw = setw + sockconn;
                }
                else {
                обслужва се i-тия клиент (ako i-тия
                клиент изпраща низ с нулева дължина,
                неговият дескриптор се премахва от setw);
                }
        }
}
```

**UDP Echo Server**

```
int mysock;
struct sockaddr_in myaddr, cliaddr;
char msgbuf[MAXLEN];
socklen_t clilen;
int msglen;
mysock = socket(PF_INET,SOCK_DGRAM,0);
myaddr.sin_family = AF_INET;
myaddr.sin_port = htons( S_PORT );
myaddr.sin_addr = htonl( INADDR_ANY );
bind(mysock, &myaddr, sizeof(myaddr));
while (1) {
        len=sizeof(cliaddr);
        msglen=recvfrom(mysock,msgbuf,MAXLEN,0,
        cliaddr, &clilen);
        sendto(mysock,msgbuf,msglen,0,cliaddr,clilen);
}
```

**RPC Programming**

```
/*
* The average procedure receives an array of real
* numbers and returns the average of their
* values. This toy service handles a maximum of
* 200 numbers. */
const MAXAVGSIZE = 200;
struct input_data {
        double input_data<200>;
};
typedef struct input_data input_data;
program AVERAGEPROG {
        version AVERAGEVERS {
                double AVERAGE(input_data) = 1;
        } = 1;
} = 22855;
```

rpcgen avg.x
produces:
1. avg_clnt.c: the stub program for our client (caller) process
2. avg_svc.c: the main program for our server (callee) process
3. avg_xdr.c: the XDR routines used by both the client and the server
4. avg.h : the header file

**Developing an RMI System**

**1)Defining the Remote Interface**

```
/* SampleServer.java */
import java.rmi.*;
public interface SampleServer extends Remote
{public int sum(int a,int b) throws RemoteException;}
```

**2)Develop the remote object and its interface**

```
/* SampleServerImpl.java */
import java.rmi.*;
import java.rmi.server.*;
import java.rmi.registry.*;
public class SampleServerImpl extends UnicastRemoteObject
implements SampleServer
{
        SampleServerImpl() throws RemoteException
        {super();}
        //Implement the remote methods
        /* SampleServerImpl.java */
        public int sum(int a,int b) throws RemoteException
        { return a + b; }
}
//Implement Server
/* SampleServerImpl.java */
public static void main(String args[])
{
        try{System.setSecurityManager(new
        RMISecurityManager());
        //set the security manager
        //create a local instance of the object
        SampleServerImpl Server = new SampleServerImpl();
        //put the local instance in the registry
        Naming.rebind(“//localhost/SAMPLE-SERVER" , Server);
        System.out.println("Server waiting.....");
}catch (java.net.MalformedURLException me)
{ System.out.println("Malformed URL: " + me.toString()); }
        catch (RemoteException re)
{System.out.println("Remote   exception: " + re.toString()); }
}
```

**3)Develop the client program**

```
import java.rmi.*;
import java.rmi.server.*;
public class SampleClient
{
        public static void main(String[] args){
        // set the security manager for the client
        System.setSecurityManager(new
        RMISecurityManager());
        //get the remote object from the registry
        try{
                System.out.println("Security Manager loaded");
                String url = "//localhost/SAMPLE-SERVER";
                SampleServer remoteObject =
                (SampleServer)Naming.lookup(url);
                System.out.println("Got remote object");
                System.out.println(" 1 + 2 = "
                +remoteObject.sum(1,2));
        }
        catch (RemoteException exc) {
System.out.println("Error in lookup: " + exc.toString()); }
catch (java.net.MalformedURLException exc) {
System.out.println("Malformed URL: " + exc.toString()); }
catch (java.rmi.NotBoundException exc) {
System.out.println("NotBound: " + exc.toString());
}        }        }
```

**4)Compile the Java source files &**
**5)Generate the client stubs and server** skeletons
**6)Start the RMI registry**
• The RMI applications need install to Registry. And the Registry must start manual by call rmiregisty.
• The rmiregistry us uses port **1099** by default.
• Bind rmiregistry to a different port: **rmiregistry<newport>**
elpis:~/rmi> **rmiregistry**
Windows: > **start rmiregistry**
**7)Start the remote server** objects
**8)Run the client**

**Жизнен цикъл на аплета:**

```
import java.awt.*;
import java.applet.Applet;
public class AppletStructure extends Applet {
        public void init() {
                System.out.println("initializing"); } // init
        public void start() {
                System.out.println("starting"); }// start public
        public void paint(Graphics g) {
                System.out.println("painting");
                g.drawString("Hello World!", 30, 30); } // paint
        public void stop() {
                System.out.println("stopping"); }// stop
}
```

**CGI модули**

**Команден интерпретатор – метод GET**

```
#!/bin/sh
# cgi1.sh
# A simple script for showing environment variable
# information passed to a CGI program.
        echo Content-type: text/plain
        echo
# Next, we want to display the arguments.
        echo argv is "$*".
        echo
# Then we show the environment variables under which the
# CGI request was made.
        echo SERVER_SOFTWARE=$SERVER_SOFTWARE
        echo SERVER_NAME=$SERVER_NAME

        . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

## Команден интерпретатор - метод POST

```
#!/bin/sh
# cgi2.sh
# A simple script for showing environment variable
# information passed to a CGI program by method POST.
        echo Content-type: text/plain
        ......................................................................
        echo CONTENT_LENGTH=$CONTENT_LENGTH
        echo The data was
        read X
        while [ “$X” != “” ]
        do
                echo $X
                read X
        done
        exit 0
```

## CGI Модул –C програма

Четене на стойностите на променливите на обкръжението от C programa.

```
char_ptr = getenv("REQUEST_METHOD");
```

**Пример:**

```
#include <stdlib.h>
#include <stdio.h>
int main(int argc, char *argv[])
{
        printf("Content-type: text/plain\n\n");
        printf("Argument number is %d\n", argc);
        printf(" SERVER_SOFTWARE=%s\n",
        getenv("SERVER_SOFTWARE"));
        printf(" SERVER_NAME=%s\n",
        getenv("SERVER_NAME"));
        printf(" GATEWAY_INTERFACE=%s\n",
        getenv("GATEWAY_INTERFACE"));
        printf(" SERVER_PROTOCOL=%s\n",
        getenv("SERVER_PROTOCOL"));
        printf(" SERVER_PORT=%s\n",
        getenv("SERVER_PORT"));
        ( . . . . . . . . . . . . . . . . . . . . . . .)
}
```

## Свободна типизация

```
<?php
        print("PHP data types<BR>");
        $var = 1 + 2;
        print("VAR is $var. <BR>");
        $var ="one " ;
        print("VAR is $var. <BR>");
?>
```

Резултат:

```
PHP data types
VAR is 3.
VAR is one .
```

## PHP arrays

```
<?php
        $note["Geometry"] = 4;
        $note["OS"] = 6;
        $note["Num methods"] = 3;
        reset($note);
        $value = current($note);
        $key = key($note);
        print ("$key = $value <br>");
        while ($value=next($note))
        {
                $key = key($note);
                print ("$key = $value <br>");
        }
?>
```

Резултат:

```
Geometry = 4
OS = 6
Num methods = 3
```

## PHP Дефиниция на клас

```
class SimpleClass [extends ParentClass]
{
        // дефиниция на член
        public $var = 'стойност по подразбиране';
        // дефиниция на конструктор
        void __construct (аргументи ) { }
        // дефиниция на метод
        public function displayVar() {
                echo $this->var;
        }
}
```

## Accessing the values of the form controls

```
<form action="foo.php" method="post">
Name: <input type="text" name="username" /><br />
Email: <input type="text" name="email" /><br />
<input type="submit" name="submit" value="Submit me!" />
</form>
<?php
        // Available since PHP 4.1.0
        echo $_POST['username'];
        echo $_REQUEST['username'];
?>
```

## PHP autorization

```
<?php
        if (!isset($_SERVER['PHP_AUTH_USER'])) {
        header('WWW-Authenticate: Basic realm="My Realm"');
                header('HTTP/1.0 401 Unauthorized');
                echo 'Text to send if user hits Cancel button';
                exit;
        } else {
echo "<p>Hello {$_SERVER['PHP_AUTH_USER']}.</p>";
echo "<p>You entered {$_SERVER['PHP_AUTH_PW']} as your
password.</p>";
        }
?>
```

## The Servlet Structure

The servlet is an Java class, which extends the base class HttpServlet.

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class <ServletName> extends HttpServlet {
//servlet methods
}
```

## Servlet example

```
public class CatalogServlet extends HttpServlet {
        private BookDB bookDB;
        public void init() throws ServletException {
        bookDB = OpenDB(“Book DB”);
        if (bookDB == null)
                throw new UnavailableException("Couldn't get
database.");
        }
}
```

## doGet method

```
public void doGet (HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {
        .....................................................................
}

String bookId = request.getParameter("Add");
        if (bookId != null) {
        .......................................................................
}
```

**An example:**

```
public class BookDetailsServlet extends HttpServlet {
        public void doGet (HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {
                // set headers before accessing the Writer
                response.setContentType("text/html");
                response.setBufferSize(8192);
                PrintWriter out = response.getWriter();
                // then write the response
                out.println("<html>" +"<head><title>+
                "TitleBookDescription"+</title></head>");
.............................................................................................................
.............................................
```

## JSP Example

```
<html>
        <head>
                <title>Sample Application JSP Page</title>
        </head>
        <body bgcolor=white>
                <CENTER>
                <img src="images/tomcat.gif">
<%= new String("<BR>Tomcat salutes you!<BR>") %>
</CENTER>
<%= "The request is sent from " +request.getRemoteHost()
%>
<%
        String queryData = request.getQueryString();
        if (queryData == null)
                out.println("<BR> No parameters were sent!");
        else
        out.println("<BR>Parameters are:" + queryData);
%>
        </body>
</html>
```

## An example of a Stateless Bean

```
//The purpose of converter is to calculate currency conversions
//between Japanese yen and Eurodollars.
//package com.sun.tutorial.javaee.ejb;
import java.math.BigDecimal;
import javax.ejb.Remote;
@Remote
public interface Converter {
        public BigDecimal dollarToYen(BigDecimal dollars);
        public BigDecimal yenToEuro(BigDecimal yen);
}

package com.sun.tutorial.javaee.ejb;
import java.math.BigDecimal;
import javax.ejb.*;
@Stateless
public class ConverterBean implements Converter {
private BigDecimal yenRate = new BigDecimal("115.3100");
private BigDecimal euroRate = new BigDecimal("0.0071");
public BigDecimal dollarToYen(BigDecimal dollars) {
        BigDecimal result = dollars.multiply(yenRate);
        return result.setScale(2, BigDecimal.ROUND_UP);
        }
public BigDecimal yenToEuro(BigDecimal yen) {
        BigDecimal result = yen.multiply(euroRate);
        return result.setScale(2, BigDecimal.ROUND_UP);
        }
}
```

## PHP connect to database

```
<?php
$con = mysql_connect("localhost","root");
$res=mysql_select_db("student");
mysql_query("SELECT * FROM notes where
fn=1001");
echo mysql_affected_rows();
?>
```

## JDBC example connect

```
import java.sql.*;
.................................
Connection conn = null;
try {
        String userName = "testuser";
        String password = "testpass";
        String url = "jdbc:mysql://localhost/test";
      Class.forName ("com.mysql.jdbc.Driver").newInstance ();
        conn = DriverManager.getConnection (url, userName,
        password);
        System.out.println ("Database connection
        established");
    } catch (Exception e) {
        System.err.println ("Cannot connect to database
        server");
}
```

## JDBC NO result Queries

```
Statement s = conn.createStatement ();
int count;
s.executeUpdate ("DROP TABLE IF EXISTS animal");
s.executeUpdate
( "CREATE TABLE animal (“
+ "id INT UNSIGNED NOT NULL AUTO_INCREMENT,"
+ "PRIMARY KEY (id)," + "name CHAR(40), category
CHAR(40))");
count = s.executeUpdate ( "INSERT INTO animal (name,
category)" + " VALUES" + "('snake', 'reptile'),"
+ "('frog', 'amphibian')," + "('tuna', 'fish'),"
+ "('racoon', 'mammal')"); s.close ();
System.out.println (count + " rows were inserted");
```

## Queries that return result(SELECT, ...)

```
Statement s = conn.createStatement ();
s.executeQuery ("SELECT id, name, category FROM animal");
ResultSet rs = s.getResultSet ();
int count = 0;
while (rs.next ()) {
        int idVal = rs.getInt ("id");
        String nameVal = rs.getString ("name");
        String catVal = rs.getString ("category");
        System.out.println ( "id = " + idVal + ",
                name = " + nameVal + ", category = " +
        catVal);
        ++count;
}
rs.close ();
s.close ();
System.out.println (count + " rows were retrieved");
```

## Entities example

```
import java.io.Serializable;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Table;
@Entity // java annotation
@Table(name=“EMPLOYEE TABLE")
public class Employee {
        @Id
        private int id;
        private String name;
        public Employee() { }
        public Employee(int id) {
                this.id = id; }
        public int getId() {
                return id; }
        public void setId(int id) {
                this.id = id; }
                public String getName() {
                return name; }
        public void setName(String name) {
                this.name = name; }
}
```

## An example - persisting a new entity

```
EntityManager em;
// set up a new entity instance
Employee person = new Employee(10);
person.setName("Miller");
// put it under the management of the entity manager
em.persist(person);
```

## An example - finding an entity by Its unique

identifier

```
EntityManager em;
// retrieve a managed entity instance
Employee person = em.find(Employee.class,
Integer.valueOf(10));
if (person != null) {
// schedule the entity for removal
em.remove(person);
}
```