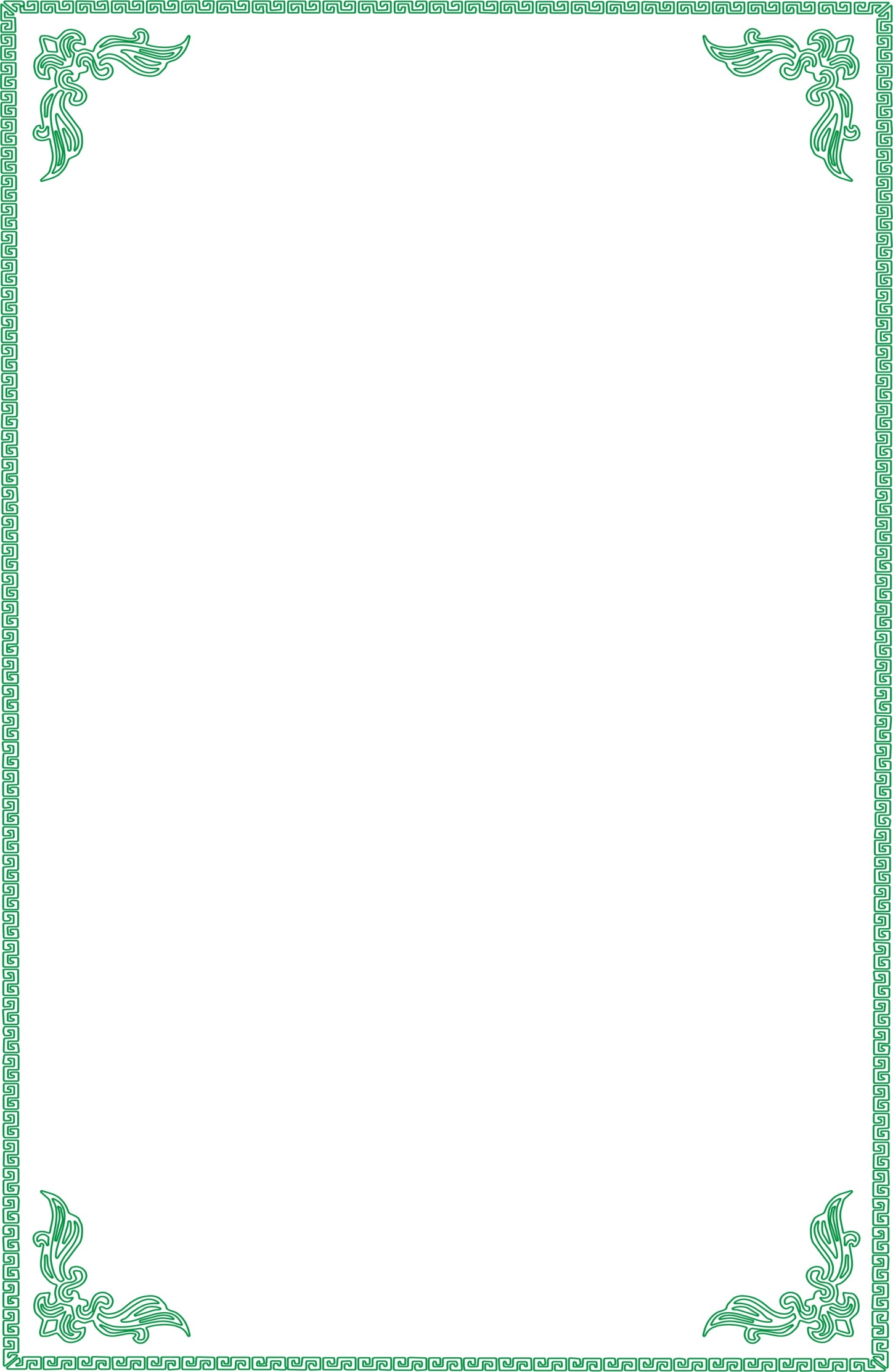
**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**VIỆN CNTT VÀ TRUYỀN THÔNG**

**\*\*\***

****

**BÁO CÁO**

**BÀI TẬP LỚN MÔN MẠNG MÁY TÍNH**

***Đề tài : Tìm hiểu mô hình lập trình mạng Socket trong Java***

*Giáo viên hướng dẫn* ***:* Nguyễn Đức Toàn**

|  |  |
| --- | --- |
|  | |
| Nguyễn Tiến Minh | 20158259 |
| Nguyễn Hiển Tường | 20158419 |
|  |  |

*Sinh viên thực hiện****:***

*Hà nội 12/2018*

**LỜI NÓI ĐẦU**

*Ngày nay, hầu như việc viết một ứng dụng để chạy trên máy đơn cục bộ không còn được ưa chuộng và thích hợp nữa. Các chương trình và ứng dụng hiện đại phải tích hợp và triệu gọi lẫn nhau trên mạng Intranet (mạng cục bộ), mạng Internet (mạng toàn cầu) và ngôn ngữ lập trình Java là một trong những lựa chọn tốt nhất để làm việc này. Java là một ngôn ngữ lập trình không đơn giản, ngoài sự nổi tiếng về bản thân ngôn ngữ, nền tảng Java còn hướng đến các ứng dụng mạng như: giao tiếp trên mạng theo mô hình khách/chủ (client/server) … So với lập trình thông thường, lập trình mạng đòi hỏi người lập trình những hiểu biết và kỹ năng chuyên sâu hơn để tạo giao tiếp và trao đổi dữ liệu giữa các máy tính với nhau.*

*Lập trình mạng là môn học mới được đưa vào giảng dạy, nên trong quá trình soạn thảo bài giảng không tránh khỏi bỡ ngỡ và thiếu sót. Chúng tôi rất mong được sự giúp đỡ, đóng góp ý kiến của các đồng nghiệp và độc giả quan tâm để lần tái bản sau cuốn sách được hoàn thiện hơn.chúng tôi xin chân thành cảm ơn!*

**MỤC LỤC**

[**LỜI NÓI ĐẦU**](#page1)[**1**](#page1)

[**MỤC LỤC**](#page2)[**2**](#page2)

[**PHẦN 1: TỔNG QUAN VỀ LẬP TRÌNH MẠNG**](#page5)[**5**](#page5)

*I.* [*Các giao thức mạng*](#page5)[*5*](#page5)

I.1. [Họ giao thức TCP/IP](#page5) [5](#page5)

I.2. [Giao thức TCP và UDP](#page6) [6](#page6)

I.3. [Dịch vụ từ phía máy chủ và khái niệm cổng (PORT)](#page7) [7](#page7)

*II.* [*Giao tiếp trên mạng theo mô hình khách/chủ (Client/Server) và khái niệm socket*](#page8)[*8*](#page8)

II.1. [Giao tiếp theo mô hình khách/chủ (Client/Server)](#page8) [8](#page8)

II.2. [Lập trình mạng thông qua Socket](#page8) [8](#page8)

II.3. [Tìm hiểu một số lớp cần thiết của gói thư viện Java.net](#page8) [8](#page8)

[**PHẦN 2:**](#page102)[**LẬP TRÌNH SOCKET**](#page102)[**102**](#page102)

[CHƯƠNG 1: LậP TRÌNH TCP SOCKET](#page102) [102](#page102)

*I.* [*Xây dựng chương trình EchoServer*](#page102)[*102*](#page102)

*II.* [*Xây dựng chương trình EchoClient*](#page103)[*103*](#page103)

[CHƯƠNG 2: LậP TRÌNH UDP SOCKET](#page105) [105](#page105)

*I.* [*Xây dựng chương trình ExchangeRateServer*](#page105)[*105*](#page105)

*II.* [*Xây dựng chương trình ExchangeRateTable*](#page106)[*106*](#page106)

[**TÀI LIỆU THAM KHẢO**](#page119)[**119**](#page119)

**Phần 1: Tổng quan về lập trình mạng**

**I. Các giao thức mạng**

**I.1. Họ giao thức TCP/IP**

Để hai hay nhiều máy có thể giao tiếp được với nhau, chúng phải dùng một ngôn ngữ chung: chẳng hạn máy này phải gửi những tín hiệu gì đến máy kia và máy kia phải gửi trả lại những tín hiệu nào để nhận biết. Trên Internet ngày nay việc hai máy có thể trao đổi được với nhau đa số đều dựa theo quy ước hay giao thức cốt lõi là TCP/IP (Transmission Control Protocol/ Internet Protocol). Theo giao thức này, mỗi máy sẽ được đặt cho một số riêng biệt gọi là địa chỉ IP (IP address) có vai trò tương tự số điện thoại, chẳng hạn máy tính có tên là [www.microsoft.com](http://www.microsoft.com/) sẽ có địa chỉ IP là 207.46.230.219. Các số IP này là duy nhất và không máy nào được trùng nhau (trên toàn thế giới). Khi bạn muốn máy của mình có địa chỉ IP để tham gia vào hệ thống Internet toàn cầu như là một máy chủ (host hay server) ta phải đăng ký với tổ chức quốc tế InterNIC (Internet Network Information Center) để nhận được một số IP riêng biệt.

Vậy tại sao vừa có địa chỉ IP lại vừa có tên riêng cho từng máy? Dùng cái nào để xác định liên lạc với một máy chủ (như ở trên, ta nên dùng [www.microsoft.com](http://www.microsoft.com/) hay số 207.46.230.219 để kết nối với một máy chủ của công ty Microsoft). Thật ra tên và địa chỉ IP là một, nhưng địa chỉ IP được ghi bằng số, còn tên của máy chủ lại được ghi bằng chữ có ý nghĩa và gần gũi hơn với con người. Với mỗi hệ thống đều có sự chuyển đổi trực tiếp từ tên vùng thành địa chỉ IP thích hợp trước khi dữ liệu được gửi đi.

Ví dụ đối với Windows có thể tham khảo hai tập tin HOSTS và LMHOSTS đây là hai tập tin văn bản (được coi như một cơ sở dữ liệu) để lưu trữ tập hợp các số IP cùng với tên tương ứng. Ta có thể tự thêm vào địa chỉ IP và tên máy chủ cách nhau bằng khoảng trắng. Khi có nhu cầu truy cập đến một máy ở xa qua giao thức mạng TCP/IP nếu gõ vào tên máy chủ thì hệ thống sẽ tự tìm địa chỉ IP tương ứng trong tập tin này:

Tập tin HOSTS

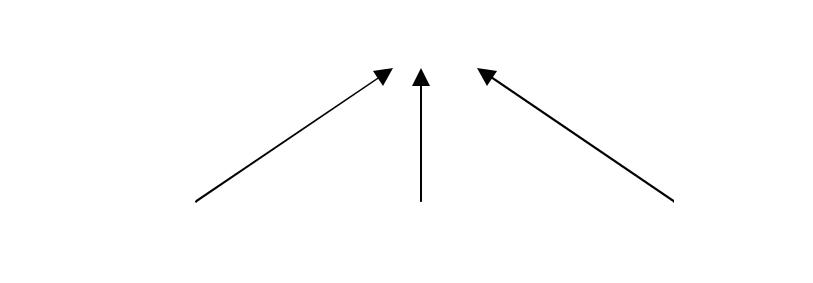
216.32.74.52 [www.myyahoo.com](http://www.myyahoo.com/)

164.71.2.70 [www.fujitsu.com](http://www.fujitsu.com/)

Tên của máy chủ còn được gọi là tên vùng (domain name) bởi vì chúng được đặt theo thứ tự phân cấp của tên lãnh thổ, vùng, tổ chức, hay tên cá nhân... mỗi nhóm phân cấp cách nhau bởi một dấu chấm (.). Công việc theo dõi sự thay đổi tên được phân phối qua Internet nhờ các máy chủ lớn DNS (Domain Name System) theo dõi các máy chủ khác trong vùng con của chúng.

Trước đây mỗi máy có thể tự mình lưu trữ một tập tin chứa phần lớn các tên và địa chỉ của những máy chủ thông dụng (như trong Windows là tập tin HOSTS và LMHOSTS), nhưng ngày nay ta không làm như vậy nữa mà đa số các tên vùng cũng như địa chỉ IP được lấy xuống từ các máy chủ DNS.

Khi đọc tên vùng của một máy chủ ta đi từ trái qua phải. Ví dụ:



**Java.sun.com**

**tên máy tính** **tên tổ chức** **tên vùng**

Nói chung là theo quy ước từ phần riêng biệt nhất đến phần chung nhất. Tuy nhiên không bắt buộc là như vậy, ta vẫn có thể đặt tên theo cách khác, không nhất thiết là chỉ gồm 3 hay 4 nhóm (ví dụ: here.is.along.name.address.co.vn là hợp lệ) bởi vì cuối cùng thì tên vùng cũng được hệ thống DNS chuyển thành địa chỉ IP mà thôi.

**I.2. Giao thức TCP và UDP**

Quá trình chuyển dữ liệu trên mạng là khá phức tạp. Chi tiết quá trình này diễn ra tương tự như trong thực tế ta gửi thư hay bưu phẩm, trước hết phải ghi rõ địa chỉ nơi đến (trường hợp này là địa chỉ IP của máy chủ), sau đó có thể gửi thông thường hoặc gửi bảo đảm (tùy theo cách gửi mà thư hay bưu phẩm có chắc chắn đến được tay người nhận hay không), người nhận sau khi nhận được có thể hồi âm trả lời là đã nhận đủ hoặc bị mất mát gì đó trong quá trình chuyển tải. Người gửi có thể gửi tiếp những phần bị mất (hoặc không cần gửi nữa)

Cách chuyển dữ liệu bảo đảm dựa vào giao thức TCP (Transmission Control Protocol), còn cách truyền không đảm bảo dựa vào giao thức UDP (User Datagram Protocol).

Giao thức TCP gửi từng gói dữ liệu đi, nơi nhận theo giao thức này phải có trách nhiệm thông báo và kiểm tra xem dữ liệu đã đến đủ hay chưa, có lỗi hay không có lỗi. Trước khi chuyển dữ liệu bao giờ cũng có sự kết nối giữa máy gửi và máy nhận. Do phải đảm bảo dữ liệu được truyền chính xác và luôn duy trì kết nối nên sử dụng giao thức TCP cần chiếm thêm một số tài nguyên của hệ thống và cách lập trình cho giao thức này hơi khó (phải thực hiện các bước kiểm tra dữ liệu theo yêu cầu của TCP). Truyền dữ liệu theo TC thường áp dụng cho các dịch vụ như truyền tập tin, các dịch vụ trực tuyến trên Internet đòi hỏi độ tin cậy cao …

Giao thức UDP ngược lại không đáng tin cậy lắm, không có sự kết nối trước nào giữa nơi gửi và nơi nhận, dữ liệu gửi đi mặc định rằng máy tính ở đầu nhận luôn ở trạng thái sẵn sàng để tiếp

đón dữ liệu gửi đến. Nếu dữ liệu gửi đến bị lỗi trong quá trình truyền hay không nhận được đầy đủ giao thức UDP cũng không có thông tin phản hồi gì lại cho nơi gửi. Tuy nhiên UDP không đòi hỏi sự chính xác cao như dịch vụ thông báo giờ, tỉ giá hay các dịch vụ gửi nhắn tin …

**I.3. Dịch vụ từ phía máy chủ và khái niệm cổng (PORT)**

Ta có thể kết nối vào Internet thông qua dịch vụ của nhà cung cấp còn gọi là ISP (như FPT, Cnet ...) bằng đường điện thoại thông qua modem. Các nhà dịch vụ này đóng vai trò như những máy chủ (server) giúp dễ dàng truy cập dữ liệu từ những vùng khác nhau trên mạng.

Khi kết nối vào máy chủ ta có thể yêu cầu máy chủ nhiều dịch vụ khác nhau, như dịch vụ truy tìm và đọc các trang Web trên Internet, dịch vụ gửi nhận e-mail, dịch vụ dò tìm hệ thống tên vùng DNS ... Mỗi dịch vụ đều có cách gửi nhận dữ liệu theo quy ước riêng. TCP và UDP chỉ chịu trách nhiệm đưa dữ liệu từ một máy tính này đến một máy tính khác, còn dữ liệu đó được gửi cho dịch vụ nào thì phải thông qua một dịch vụ nữa gọi là cổng (hay port). Mỗi chương trình dịch vụ sẽ sử dụng một cổng khác để truy xuất thông tin. Cổng là một số nguyên dương có giá trị từ 1 đến 16383

Máy chủ (server) sẽ quy định cổng được sử dụng cho mỗi loại dịch vụ. Thông tin giữa máy khách (client) và máy chủ (server) phải sử dụng cổng tương ứng nhau thì mới trao đổi được với nhau. Tuy nhiên, hầu hết các chương trình dịch vụ nổi tiếng hiện nay đều có quy định chuẩn cổng dành riêng cho mình như:

Dịch vụ cổng (port)

FTP21

HTTP80

Telnet23

Finger79

SMTP25

Nếu tự xây dựng một ứng dụng làm dịch vụ trên máy chủ ta phải chọn cho mình một số cổng có các giá trị khác với những giá trị cổng mà những dịch vụ nổi tiếng khác đã sử dụng.

**II. Giao tiếp trên mạng theo mô hình khách/chủ (Client/Server) và khái niệm socket.**

**II.1. Giao tiếp theo mô hình khách/chủ (Client/Server)**

Có rất nhiều dịch vụ hỗ trợ trên Internet như e-mail, nhóm tin (newsgroup), chuyển tập tin (file transfer), đăng nhập từ xa (remote login), truy tìm các trang Web ... Những dịch vụ này được tổ chức và kiến trúc theo mô hình khách/chủ (mô hình Client/Server). Các chương trình ở máy khách (client) như trình duyệt (Web Browser) hay chương trình gửi nhận e-mail sẽ tạo ra kết nối (connection) với một máy chủ ở xa (server) sau đó gửi các yêu cầu đến máy chủ, các chương trình dich vụ trên máy chủ như Web server hay Mail server ... sẽ xử lý những yêu cầu này và gửi kết quả ngược về cho máy khách (chẳng hạn Web theo địa chỉ mà máy khách đưa đến còn Mail server thì lưu giữ và gửi về cho máy khách những bức e-mail mới). Thông thường một dịch vụ trên máy chủ phục vụ cho rất nhiều khách.

**II.2. Lập trình mạng thông qua Socket**

Như vậy trước khi yêu cầu một dịch vụ trên máy chủ thực hiện điều gì đó, máy khách (client) phải có khả năng kết nối được với máy chủ. Quá trình kết nối này được Java thực hiện thông qua một cơ chế trừu tượng hóa gọi là Socket (tạm gọi là “cơ chế ổ cắm”). Kết nối giữa máy khách và máy chủ tương tự như việc cắm phích điện vào ổ cắm điện. Máy khách thường được coi như phích cắm điện còn máy chủ được coi như ổ cắm điện, một ổ cắm có thể cắm vào đó nhiều phích điện khác nhau cũng như một máy chủ có thể kết nối và phục vụ cho rất nhiều máy khách.

Nếu kết nối socket thành công thì máy khách và máy chủ có thể trao đổi dữ liệu với nhau thực hiện các yêu cầu về dịch vụ trên máy chủ. Việc kết nối theo cơ chế socket cần biết hai thông tin chủ yếu đó là địa chỉ của máy cần kết nối và số hiệu cổng của chương trình dịch vụ. Java cung cấp lớp Socket (thường được dùng như “phich cắm điện” cho máy khách) và lớp ServerSocket (thường được dùng như “ổ cắm điện” đặt trên máy chủ). Hai lớp này được đặt trong gói thư viện Java.net.

**II.3. Tìm hiểu một số lớp cần thiết của gói thư viện Java.net**

**a. Lớp InetAddress**

Vì địa chỉ Internet theo số IP và theo tên rất thường dùng khi kết nối vào mạng cho nên Java xây dựng hẳn một lớp InetAddress dành riêng cho việc quản lý địa chỉ theo tên và theo số. Lớp InetAddress cung cấp các phương thức static thông dụng nhất dùng để chuyển đổi và truy xuất

địa chỉ IP (không có phương thức khởi dựng cho lớp này). Thường ta sẽ quan tâm đến các phương thức sau:

* public static InetAddress getLocalHost ()

throws UnknownHostExceptiongetByName

Trả về đối tượng InetAddress là địa chỉ của máy cục bộ (local host)

* public static InetAddress getByName (String host) throws UnknownHostException

Phương thức này nhận địa chỉ của một máy bằng kiểu chuỗi String và trả về đối tượng kiểu InetAddress thay mặt cho địa chỉ máy này.

* public static InetAddress[] getAllByName (String host) throws UnknownHostException

Phương thức này nhận địa chỉ của một máy bằng kiểu chuỗi và trả về tất cả các đối tượng InetAddress thay mặt cho địa chỉ máy này.

* public byte[] getAddress()

Trả về địa chỉ IP của đối tượng InetAddress dưới dạng một dãy các byte. Vị trí byte cao

nhất nằm ở byte 0.

* public String getHostAddress()

Trả về địa chỉ IP của đối tượng InetAddress dưới dạng một chuỗi được dịnh dạng phân

làm 4 nhóm %d.%d.%d.%d (Ví dụ: “172.16.11.12”).

Dưới đây là ví dụ cho thấy cách dùng lớp InetAddress để lấy về các thông tin của địa chỉ máy chủ:

Ví dụ 1-0 AddrLookupApp.java

import java.net

**b. Lớp Socket**

Lớp Socket dùng tạo kết nối từ phía khách với máy chủ thường được khởi dựng bằng các phương thức sau:

* public Socket (String host, int port)

throws UnknownHostException, IOException

Tạo ra một socket để kết nối máy có tên theo địa chỉ host và số cổng port.

* public Socket(InetAddress address, int port) throws IOException

Tạo ra một socket kết nối địa chỉ là đối tượng InetAddress và số cổng port.

Trang 9

* Public Socket(String host, int port, boolean stream) throws IOException

Tạo ra một socket kết nối theo địa chỉ host và số cổng port tham số stream cuối cùng để quy định kết nối theo TCP (stream=true) hay UDP (stream=false). Tuy nhiên nếu áp dụng để tạo socket cho giao thức UDP nên sử dụng lớp thay thế là DatagramSocket.

Các phương thức khác hỗ trợ cho lớp Socket từ phía máy khách bao gồm:

* InputStream get InputStream() throws IOException

Lấy về luồng nhập để máy khách có thể nhập dữ liệu trả về từ phía máy chủ.

* OutputStream get OutputStream() throws IOException
* InetAddress get InetAddress()
* int getPort()

Lấy về số cổng dùng kết nối của máy chủ.

Ví dụ đoạn mã sau sẽ thực hiện kết nối với máy chủ có địa chỉ “my.testing.server” và mở ra hai luồng xuất nhập để đọc và gửi thông tin đến máy chủ theo số cổng 1234.

try{

Socket me=new Socket(“my.testing.server”, 1234);

* *Luồng nhập thông tin để trả về máy chủ từ phía kết nối* DataInputstream in = new DataInputStream(me.getInputStream());
* *Luồng xuất để ghi thông tin gửi đến máy chủ*

DataOutputStream out = new DataOutputStream(me.getOutputStream()); Catch (Exception e) {

System.out.println(e);

}

**c. Lớp ServerSocket**

Lớp ServerSocket dùng tạo kết nối từ phía máy chủ với các máy khách. Đối tượng ServerSocket được tạo ra trên máy chủ và lắng nghe những kết nối từ phía máy khách gửi đến theo một số cổng địng trước. Đối tượng ServerSocket được khởi dựng từ phương thức sau:

* public ServerSocket(int port) throws IOException

**port** là sốhiệu cổng màđối tượng ServerSocket phải lắng ngheđểnhận biết những kết nối từphía máy khách gửi đến

Để chờ đợi kết nối từ các máy khách gửi đến đối tượng ServerSocket thường nhờ đến phương thức accept như sau:

* Socket accept () throws IOException

Trang 10

Phương thức này thực sự dừng lại chờ đợi cho đến khi nhận được thông tin kết nối sẻ trả về đối tượng socket của máy khách nơi có yêu cầu nối vào máy chủ.

Cuối cùng máy chủ có thể cắt đứt mọi kết nối bằng cách gọi phương thức close của đối tượng ServerSocket

* Public void close () throws IOException

Ví dụ đoạn mã sau sẽ tạo một đối tượng ServerSocket trê máy chủ luôn lắng nghe kết nối từ phía máy khách gửi đến qua số cổng 1234

Try {

ServerSocket server = new ServerSocket (1234);

Socket client;

* + - Chương trình sẽ dựng lại ở đây để chờ đợi sự kết nối client = server.accept ();
  + Có một kết nối gửi đến từ phía máy khách System.out.println(“Accept connect”);
  + Xử lý các yêu cầu về dịch vụ
  + …
* Cắt đứt các kết nối

client.close ();

server.close ();

catch (Exception e) {

System.out.println (e);

}

**d. Lớp DatagramSocket**

Lớp này được dùng để chuyển đi một gói dữ liệu (biểu diễn bằng đối tượng DatagramPackage) theo giao thức UDP. Dữ liệu được gửi đi không bảo đảm được nhận đầy đủ và có thể bị lỗi trên đường truyền (cơ chế dùng DatagramSocket không an toàn bằng lớp Socket). Dưới đây là một số phương thức dùng của lớp DatagramSocket:

* public DatagramSocket () throws SocketException Phương thức khởi dựng để tạo kết nối UDP
* public DatagramSocket (int port) throws SocketException

Phương thức khởi dựng để tạo kết nối UDP với số hiệu cổng port

* public void synchronized send (DatagramSocket p) throws IOException Gửi gói dữ liệu đi.
* Public void synchronized receive (DatagramSocket p) throws IOException

Trang 11

Nhận gói dữ liệu về.

* public void synchronized close () Đóng kết nối.

**e. Lớp DatagramPackage**

Lớp này dùng cho một gói dữ liệu gửi đi trên mạng theo kết nối DatagramSocket. Một gói có thể chứa các thông tin như dữ liệu, chiều dài gói, các địa chỉ IP và số cổng mà từ đó các gói dữ liệu được gửi đi. Dưới đây là một số phương thức hữu dụng của lớp DatagramPackage này:

* *public DatagramPackage(byte buff[], int len)*

Phương thức khởi tạo ra gói có dữ liệu chứa trong bộ đệm *buff[]* và chiều dài gói dữ liệu là *len.*

* *public DatagramPackage (byte buff[], int len, InetAddress iaddr, int port)*

Phương thức khởi tạo ra gói có dữ liệu chứa trong bộ đệm *buff[]* cùng với chiều dài vùng đệm muốn lấy, địa chỉ máy đích và số hiệu cổng.

* *public InetAddress getAddress()* Trả về địa chỉ chứa trong gói dữ liệu.
* *public byte[] getData()*

Trả về dữ liệu thật sự chứa trong gói.

* *public int getLength()*

Trả về kích thước hay chiều dài gói dữ liệu.

* *public int getPort()*

Trả vể số hiệu cổng chứa trong gói dữ liệu.

**f. Lớp URL**

URL(Uniform Resource Locator) là địa chỉ định vị tài nguyên trên mạng, thường một một URL như đã đề cập bao gồm 3 phần: phần nghi thức(protocol), phần địa chỉ hay tên máy chủ(host name), và phần chỉ định tên tập tin hay tài liệu muốn lấy từ máy chủ về.

Java đóng gói tất cả những đặc điểm này vào một lớp URL. Đối tượng URL được tạo ra bằng một trong những phương thức khởi tạo sau:

* *public URL(String spec) throws MalformedURLException* Tạo một đối tượng URL từ địa chỉ định vị là một chuỗi.
* *public URL(String protocol, String host, int port, String file) throws MalformedURLException*

Trang 12

Tạo một địa chỉ định vị tuyệt đối với đầy đủ nghi thức(protocol), máy chủ(server), cổng(port), đường dẫn(file) tới tập tin cần lấy trên máy chủ.

* *public URL(String protocol, String host, String file) throws MalformedURLException*

Tạo một địa chỉ định vị tuyệt đối với đầy đủ nghi thức(protocol), máy chủ(server), đường dẫn(file) tới tập tin cần lấy trên máy chủ(bỏ qua thành phần định vị cổng giao tiếp, nếu truy tìm trang web theo nghi thức http thì cổng chương trình web server mặc định là 80).

Các phương thức hỗ trợ khác dùng cho lớp URL là:

* *public final Object getContent() throws IOException*

lấy về nội dung mà kết nối theo địa chỉ URL có được.

* *String getFile()*

Lấy về tên tập tin hay tài liệu nằm trong chuỗi địa chỉ URL có được.

* *String getHost()*

Lấy tên máy chủ(thường là thành phần thức 2 của chuỗi URL)

* *String getPort()*

Lấy về số hiệu cổng.

* *String getProtocol()*

Lấy về tên giao thức(thường là thành phần đầu tiên trong chuỗi URL)

* *String getRef()*

Lấy về nội dung chuỗi tham khảo thêm trong chuỗi URL(được đặt sau dấu # của chuỗi)

* *Public final InputStream openStream() throws IOException*

Mở luồng nhập để đọc thông tin trả về từ máy chủ

Ví dụ đoạn mã sau đây dùng để lấy về nội dung trang web index.htm từ máy chủ java.sun.com:

try{

* *Mở kết nối đến trang Web theo địa chỉ định vị URL* URL o = new URL(“http://java.sun.com/index.htm”);
* *Tạo luồng nhập để đọc nội dung trang Web trả về từ máy chủ*

BufferedReader inStream = new BufferedReader(new InputStreamReader(o.openStream()));

* *In nội dung trang Web index.htm ra màn hình* String line;

while((line = in.readLine()) != null) {

Trang 13

System.out.println (line);

}

}catch (Exception e) {

* *Quá trình mở và kết nối với trang web bị lỗi* System.out.println (e) ;

}