ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG VIỆT - HÀN
Vietnam - Korea University of Information and Communication Technology

BÀI GIẢNG
LẬP TRÌNH MẠNG

PGS.TS.Huỳnh Công Pháp; Nguyễn Anh Tuấn; Lê Tân;
Nguyễn Thanh Cẩm; Hoàng Hữu Đức

Khoa Khoa học máy tính

Bài 7. Giao thức HTTP

(Hypertext Transfer Protocol)

Tổng quan

- Bài này trình bày tổng quan một số giao thức ứng dụng Internet phổ biến.
- Một giao thức ứng dụng cho phép việc truyền thông giữa 2 ứng dụng.
- Khi chúng ta check email, lướt Web, chơi games, hay download files trên mạng Internet, ứng dụng mạng sử dụng các giao thức ứng dụng tương ứng cho việc truyền thông. Ví dụ: giao thức POP3, HTTP, FTP,…

Tài liệu đặc tả giao thức ứng dụng

- Các giao thức Internet phổ biến sử dụng hiện nay được đăng tải như một tài liệu RFC (Request For Comment (RFC).
- Mỗi RFC chi tiết một giao thức ứng dụng, và được gán một số để nhận biết (vd, RFC 1945 cho HTTP/1.0).
- Các tài liệu RFC hết sức chi tiết và chứa tất cả các thông tin yêu cầu cho việc lập trình với giao thức này

Tìm kiếm tài liệu RFC

- Các tài liệu RFC tồn tại dưới dạng văn bản và miễn phí trên internet.
- Chúng ta có thể tìm tại http://www.rfc-editor.org/rfc.html.
- Web site RFC Editor cho phép người dùng tìm tài liệu RFCs bằng số hay bằng từ khóa, tác giả, hay tiêu đề.
- Ví dụ,
  - Để tìm tài liệu RFC 2324, chúng ta có thể gỏ "2324".
  - Để tìm FTP, chúng ta có thể gõ "File Transfer Protocol."

Giao thức HTTP (HyperText Transfer Protocol)

Tổng quan

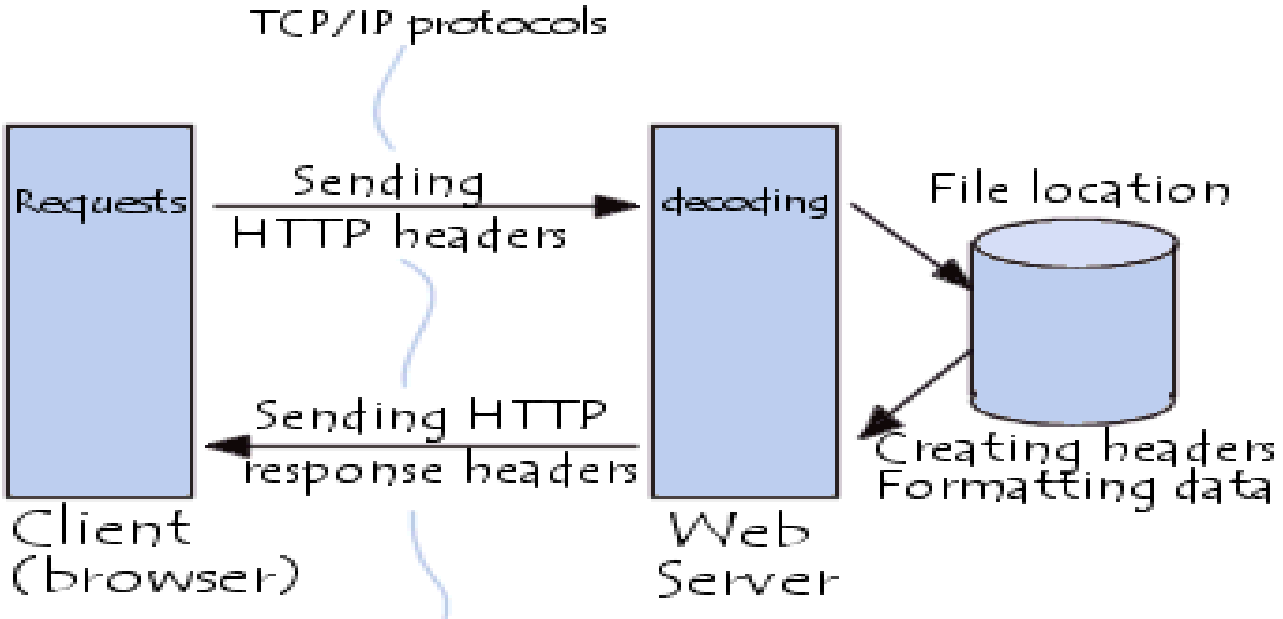
Nếu không có HTTP, sẽ không có World Wide Web, không có thương mại điện tử, cũng không có sự bùng nổ Internet…

- Giao thức này sử dụng phổ biến nhất từ năm 1990
- Mục đích của giao thức HTTP cho phép truyền files (chủ yếu là định dạng HTML) giữa trình duyệt (browser) và một Web server được định vị bởi URL.
- HTTP là giao thức ở tầng ứng dụng sử dụng TCP như cơ chế truyền.

- Trình duyệt sử dụng HTTP để yêu cầu files hay yêu cầu nội dung động sinh ra từ các ứng dụng server.
  - Tài liệu Hypertext (siêu văn bản) chứa các hyperlinks, liên kết đến các tài liệu hypertext và các files khác.
  - World Wide Web là tập hợp các tài liệu hypertext, lưu trữ ở các Web server khác nhau và được truy cập bởi rất nhiều Web browsers và HTTP clients.

Cơ chế hoạt động của HTTP

- Khi một HTTP client (Web browser) cần truy cập một, nó thiết lập một kết nối TCP đến Web server (mặc định sử dụng TCP cổng 80).
- Client gửi một yêu cầu (request) một file nào đó, và nhận một phản hồi (response) HTTP.
- Phản hồi gồm mã trạng thái (thành công/thất bại), thông tin header HTTP như chiều dài nội dung và kiểu, nội dung file.



Web Clients

- Một Web client, kết nối đến Web server qua một TCP socket, sẽ gửi một HTTP request và đọc phản hồi từ server (server response).
- Có 3 kiểu request mà một ứng dụng client gửi đến Web server:
    1. GET
    2. HEAD
    3. POST
- HTTP request phổ biến và đơn giản là GET, gọi một nguồn tài nguyên từ Web server.
- Những request phức tạp có thể là HEAD và POST.

Phương thức GET Request

- GET request chứa 2 đối số:
  - đường dẫn tài nguyên và phiên bản (version) của HTTP sử dụng.
- Ví dụ:

  GET /index.html HTTP/1.0

  GET /images/banner.gif HTTP/

Phương thức HEAD Request

- Đôi khi một client chỉ cần thông tin về tài nguyên mà không thực sự cần nguồn tài nguyên đó.
  - Ví dụ, đã có một file được tải và lưu tạm thời (cached) ở máy client, client đó chỉ cần biết thông tin về version hiện tại của file đó ở server để xem file đó có thay đổi chưa. Nếu file đã thay đổi thì mới yêu cầu tải về máy, ngược lại thì không.
- Một Head request chứa các đối số giống như GET request, và HTTP response sẽ trả lại các thông tin về tài nguyên được lưu trữ ở các trường header.
  - Ví dụ một HEAD request:
    - HEAD /files/averybigfile.zip HTTP/1.0

Phương thức POST Request

- Các scripts CGI (Common Gateway Interface), và sau đó, các ứng dụng phía server (servlet, ASP) cho phép người dùng tương tác với các ứng dụng phía server này.
- Do vậy, phải có một cách cho phép trình duyệt truyền thông tin từ người dùng đến Web server.
- Có thể dùng phương thức GET request để mã hóa các đối số truyền đến Web server.
- Ví dụ: /view/TM/ExactSearch?projName=EOLSS&source=en&tran=fr&sourcetext="how%20are%20you"
- Tuy nhiên, chiều dài của URL bị hạn chế. Do đó, nhiều lúc chúng ta không thể truyền hết thông tin người dùng đến Web Server.
- Vì thế, cách tốt hơn là chúng ta sử dụng phương thức POST request, cho phép client gửi nhiều thông tin hơn đến Web Server.
- POST request có định dạng tương tự như GET request. Sau khi các trường header đã được gửi, client gửi thân của request, chứa các đối số và các thông tin khác.

POST request gửi dữ liệu đến Server

POST /cgi-bin/information.pl HTTP/1.0 ← **POST request**

Content-type: application/x-www-form -urlencoded ← **Header**
Content-length: 21
User-Agent: SomeBrowser/1.0

Name=David%20Reilly&answer=yes ← **Thân của request**

HTTP/1.0 200 OK ← HTTP Response
Last-Modified: Monday, 27-Dec-99 22:14 GMT
Content-Type: text/html
Content-Length: 4855
HTTP Server Response
<HTML>
<HEAD>
<TITLE> Thanks for your feedback </TITLE>
</HEAD>
<BODY>
..... // BODY GOES HERE
</BODY>
</HTML>

Web Servers

- Khi một client kết nối đến một Web server, client sẽ gửi một HTTP request và server sẽ đọc và xử lý.
- Server trả lại một phản hồi gồm các thành phần sau:
  - Dòng trạng thái (Status Line), với mã số trạng thái và một thông điệp ứng với mã số này.
  - Header của phản hồi, gồm 1 hoặc nhiều trường header
  - Nội dung của phản hồi (thường trang HTML do ứng dụng Server sinh ra)

Dòng trạng thái (Status Line)

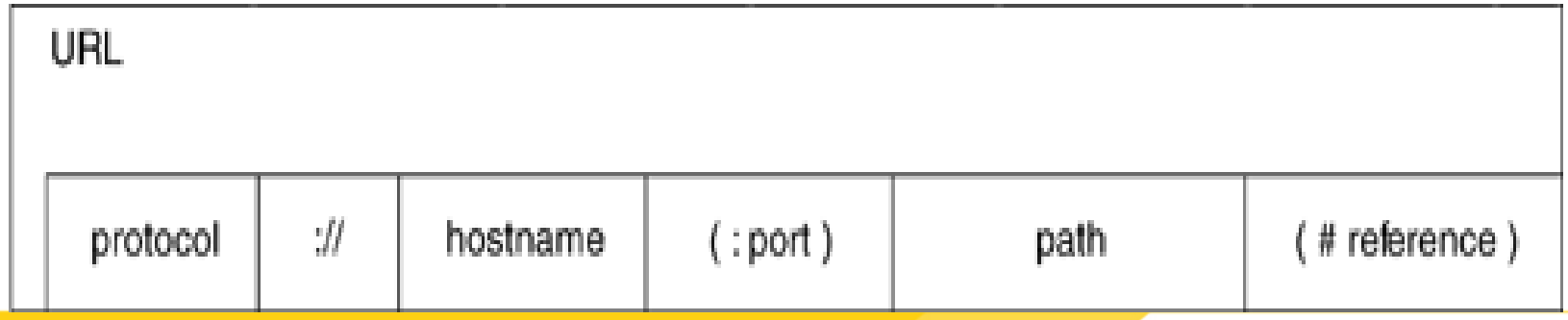
- Chỉ định request có thành công hay không
- Sử dụng một mã 3 chữ số
  - Chữ số đàu tiên ứng với nhóm lỗi
  - 2 chữ số còn lại chỉ định cụ thể điều kiện lỗi
- Bảng lỗi

Status Group	Error Group Name
1xx	Informational
2xx	Successful
3xx	Redirection
4xx	Client Error
5xx	Server Error

Lập trình với giao thức HTTP

Chúng ta có thể sử dụng Java để lập trình HTTP

- Sử dụng lớp URL để đại diện cho địa chỉ URL.
- Các URL có thể chỉ đến files, Web sites, ftp sites, newsgroups, địa chỉ email, và các tài nguyên khác.
  - Ví dụ các URLs:
    - http://vnexpress.net/GL/Home/
    - ftp://records.area51.mil/roswell/subjects/autopsy/
    - telnet://localhost:8000/
    - mailto:president@whitehouse.gov?subject=My%20Opinion
  - Định dạng URL



Lớp URL

- Lớp URL cung cấp các hàm constructors khác nhau để tạo đối tượng URL.
  - new URL("http", "www.gamelan.com", "/pages/Gamelan.net.html");
  - new URL("http://www.gamelan.com/pages/Gamelan.net.html");
  - new URL("http", "www.gamelan.com", 80, "pages/Gamelan.html");
- Lớp này cũng cung cấp các phương thức cho phép truy vấn đối tượng URL.
  - openStream() trả lại một stream cho phép đọc nội dung của URL.
  - openConnection() mở kết nối đến đối tượng URL

Ví dụ đoạn mã Java: tạo đối tượng URL

```
URL myURL = new URL("http://www.gamelan.com/pages/Gamelan.net.html");

System.out.println ("Protocol : " + myURL.getProtocol() );

System.out.println ("Hostname : " + myURL.getHost() );

System.out.println ("Port : " + myURL.getPort() );

System.out.println ("Filename : " + myURL.getFile() );

System.out.println ("Reference: " + myURL.getRef() );
```

Nhận tài nguyên sử dụng lớp URL

- Có hai phương thức cho phép nhận tài nguyên từ xa:
  - InputStream URL.openStream()
  - URLConnection URL.openConnection();
- Ví dụ mã Java

```
URL myURL = new URL ( URLString );
InputStream in = myURL.openStream();
BufferedInputStream bufIn = new BufferedInputStream(in);
while ((data = bufIn.read())!=-1) System.out.print ( (char) data);
```

Lớp URLConnection

- Lớp URLConnection được dùng để gửi các HTTP requests và đọc các HTTP responses.
- URLConnection có các phương thức cho phép nối kết đến các Web server, để thiết lập các trường header của request, đọc các trường header của response, và tất nhiên, đọc nội dung của tài nguyên.
- Ví dụ tạo đối tượng URLConnection

```
URL url = new URL ( some_url );
URLConnection connection = url.openConnection();
```

Ví dụ Java nhận tài nguyên với lớp URLConnection

```
java.net.URL myURL = new URL ( some_url );
    // Create a URLConnection object, for this URL
    // NOTE : no connection has yet been established
URLConnection connection = myURL.openConnection();
   // Now open a connection
connection.connect();
InputStream in = connection.getInputStream();
   // Buffer the stream, for better performance
BufferedInputStream bufIn = new BufferedInputStream(in);
 //read bufIn
 .....
```

Lớp HttpURLConnection

- Tự tìm hiểu về lớp này

- **Sinh viên tự tìm hiểu:**
  - **Lập trình giao thức truyền tin đơn giản SMTP (**Simple Mail Transfer Protocol)
    - **Tạo chương trình SMTP Client**
  - Lập trình giao thức nhận tin POP3 (Post Office Protocol)
    - **Tạo chương trình POP3 Client**

Bài tập lập trình giao thức HTTP

- Tạo một browser đơn giản
  - Browser này cho phép người dùng gõ URL yêu cầu tài nguyên của một trang Web nào đó.
  - Browser cũng cho phép thực hiện 3 loại request
    - GET
    - POST
    - HEAD
  - Nếu phương thức request là POST hoặc GET, thì Browser này phân tích trang HTML trả lại và hiển thị chiều dài, số lượng tag <p>, , , <imag> của trang HTML.
    - Nếu SV nào có khả năng tạo Browser có khả năng hiểu và biểu diễn trang HTML thì càng tốt.
  - Nếu phương thức request là HEAD, Browser hiển thị thông tin về tài nguyên

Thank your listening