GIAO TI**Ế**P V**Ớ**I WEB SERVER

LẬP TRÌNH MẠNG CĂN BẢN

NỘI DUNG

▪ Giới thiệu

▪ HTTP, Web Server, Web Client

▪ WebRequest & WebClient & HttpClient

▪ HttpListener

Chương 4: Truyền thông với Web server **2**

GIỚI THIỆU

▪ Cách lấy dữ liệu từ Web và sử dụng vào mục đích khác ▪ Tại sao ứng dụng cần giao tiếp với website?

▪ Kiểm tra các bản cập nhật, sửa lỗi, nâng cấp

▪ Lấy thông tin về dữ liệu được cập nhật hằng giờ ▪ Tự động truy vấn dữ liệu từ các dịch vụ điều hành bởi bên thứ 3

▪ Xây dựng search engine

▪ Cache các trang web để truy cập nhanh hơn

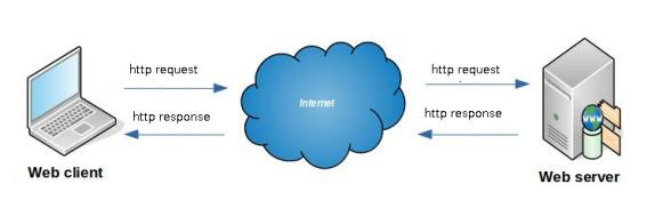
Chương 4: Truyền thông với Web server **3**

MÔ HÌNH HOẠT ĐỘNG

▪ Web Server không phải Website

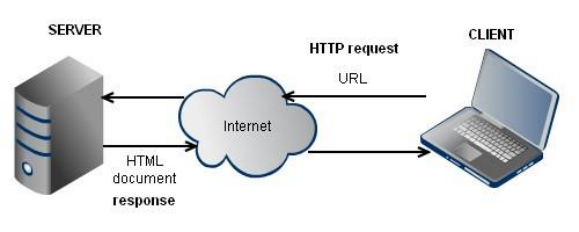
▪ Web Client thường là trình duyệt (Browser)

▪ HTTP là giao thức sử dụng để giao tiếp giữa Web Server và Web Client

Chương 4: Truyền thông với Web server **4**

HTTP

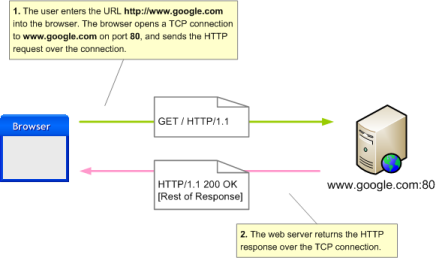
Hoạt động trên giao thức TCP/IP port 80

Port 80

Chương 4: Truyền thông với Web server **5**

HTTP

Hoạt động trên giao thức TCP/IP port 80

Gồm 2 loại thông điệp: HTTP Request & HTTP Response 

Chương 4: Truyền thông với Web server **6**

HTTP REQUEST

▪ Có 2 dạng phổ biến: GET và POST

▪ Còn có HEAD, OPTIONS, PUT, DELETE, TRACE

Chương 4: Truyền thông với Web server **7**

HTTP REQUEST

▪ Dạng đơn giản nhất:

GET /

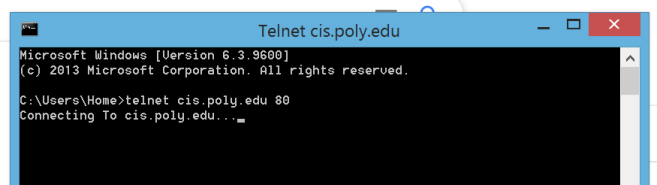
<enter><enter>

▪ Lưu ý: một số server cần xác định DNS name trong lệnh GET

Chương 4: Truyền thông với Web server **8**

VÍ DỤ: HTTP REQUEST

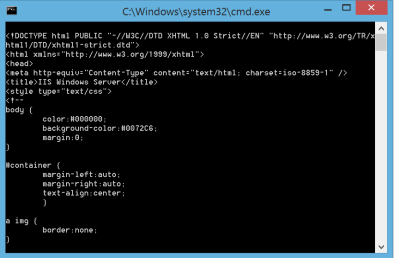
▪ Mở kết nối tới port 80 của 1 website

▪ Gõ GET /

Chương 4: Truyền thông với Web server **9**

VÍ DỤ: HTTP REQUEST

▪ Gõ GET / => Gửi thông điệp HTTP Request GET ▪ Kết quả nhận được:



Chương 4: Truyền thông với Web server **10**

HTTP REQUEST GET

Dòng yêu cầu

(các lệnh GET, POST, HEAD)

**GET /index.html HTTP/1.1\r\n Host: www-net.cs.umass.edu\r\n User-Agent: Firefox/3.6.10\r\n**

Ký tự xuống dòng Ký tự về đầu dòng

Các dòng header

Ký tự xuống dòng, về đầu dòng mới chỉ điểm cuối cùng

của thông điệp

**Accept: text/html,application/xhtml+xml\r\n Accept-Language: en-us,en;q=0.5\r\n Accept-Encoding: gzip,deflate\r\n Accept-Charset: ISO-8859-1,utf-8;q=0.7\r\n Keep-Alive: 115\r\n**

**Connection: keep-alive\r\n**

**\r\n**

Chương 4: Truyền thông với Web server **11**

STANDARD HTTP REQUEST HEADERS

| HTTP header | Ý nghĩa |
| --- | --- |
| Accept | Xác định kiểu MIME nào được chấp nhận cho response. \*/\* chỉ thị cho chấp nhận tất cả. Type/\* chỉ thị các kiểu con của type đó. |
| Accept-Charset | Xác định các character set được chấp nhận trong response. Nếu client phát Accept-Charset: iso-8859-5 thì server biết rằng client không hiển thị được các ký tự tiếng Nhật |

Chương 4: Truyền thông với Web server **12**

STANDARD HTTP REQUEST HEADERS (CNT)

| HTTP header | Ý nghĩa |
| --- | --- |
| Accept-Encoding | Xác định client có thể quản lý dữ liệu nén. Trong ví dụ trên cho biết trình duyệt hiểu được chuẩn nén GZIP |
| Accept-Language | Xác định ngôn ngữ thích hợp cho người dùng, có thể liên quan vị trí địa lý, ví dụ en-gb chỉ thị United Kingdom |
| Authorization | Cung cấp chứng thực giữa client và server |
| Host | Chỉ địa chỉ IP của server có thể dùng, có thể khác với địa chỉ IP đích nếu phải đi qua proxy. |
| If-Modified-Since | Cho biết trang web không cần trả về nếu không có thay đổi từ ngày xác định. Điều này cho phép cơ chế cache làm việc hiệu quả hơn. Ví dụ: If-Modified-Since: Sat, 29 Oct 1994 19:43:31 GMT |

Chương 4: Truyền thông với Web server **13**

STANDARD HTTP REQUEST HEADERS (CNT)

| HTTP header | Ý nghĩa |
| --- | --- |
| Proxy-Authorization | Cung cấp chứng thực giữa client và proxy |
| Range | Cung cấp cơ chế lấy một phần trang web dựa trên vùng byte. Ví dụ: bytes=500-600,601-999 |
| Referer | Cho biết trang client vừa xem |
| TE | Transfer encoding (TE) cho biết phần mở rộng encoding có thể chấp nhận |
| User-Agent | Chỉ kiểu trình duyệt client đang dùng |
| Content-Type | Dùng trong các POST request, chỉ kiểu MIME của dữ liệu được post lên, thông thường là  application/x-www-form-urlencoded |
| Content-Length | Dùng trong các POST request, chỉ độ dài của dữ liệu (đi sau 2 dòng trống) |

Chương 4: Truyền thông với Web server **14**

HTTP POST REQUEST

POST / HTTP/1.1

Content-Type: application/x-www-form-urlencoded Content-Length: 17

myField=some+text

Chương 4: Truyền thông với Web server **15**

HTTP RESPONSE

Khi server nhận được một HTTP request, trả về trang yêu cầu cùng với HTTP header

Chương 4: Truyền thông với Web server **16**

HTTP RESPONSE

Dòng trạng thái

(giao thức

mã trạng thái

cụm từ trạng thái)

Các dòng

header

Dữ liệu, ví dụ,

tập tin HTML

được yêu cầu

**HTTP/1.1 200 OK\r\n**

**Date: Sun, 26 Sep 2010 20:09:20 GMT\r\n máy chủ: Apache/2.0.52 (CentOS)\r\n**

**Last-Modified: Tue, 30 Oct 2007 17:00:02 GMT\r\n ETag: "17dc6-a5c-bf716880"\r\n**

**Accept-Ranges: bytes\r\n**

**Content-Length: 2652\r\n**

**Keep-Alive: timeout=10, max=100\r\n**

**Connection: Keep-Alive\r\n**

**Content-Type: text/html; charset=ISO-8859-1\r\n \r\n**

**data data data data data ...**

Chương 4: Truyền thông với Web server **17**

HTTP RESPONSE

| HTTP response  header | Ý nghĩa |
| --- | --- |
| ETag | Dùng kết hợp với If-suffixed HTTP requests |
| Location | Dùng để điều hướng (redirect) sang trang web khác, kết hợp với HTTP 3xx responses |
| Proxy-Authenticate | Cung cấp chứng thực giữa client và proxy |
| Server | Chỉ phiên bản và vendor của server. Ví dụ: IIS chạy trên WindowsXP |
| WWW-Authenticate | Cung cấp chứng thực giữa client và proxy |
| Content-Type | Chỉ kiểu MIME của nội dung trả về. Ví dụ: HTML |
| Content-Length | Chỉ độ dài của dữ liệu (đi sau 2 dòng trống). Server sẽ đóng kết nối sau khi gửi tất cả dữ liệu, do đó không cần thiết xử lý lệnh này |
| Set-Cookie  Chương 4: Truyền thông với Web server | Thiết lập một cookie trên client. Cookie là một file nhỏ ghi trên client. Mỗi cookie có tên và giá trị. Ví dụ: tên cookie là ASPSESSIONIDQGGQQFCO |

**18**

HTTP RESPONSE

| HTTP response  code range | Ý nghĩa |
| --- | --- |
| 100–199 | Thông tin: Request đã được nhận, tiếp tục xử lý |
| 200–299 | Thành công: Thao tác đã nhận thành công, hiểu được và chấp nhận |
| 300–399 | Điều hướng: Phải thêm thao tác để hoàn thành request |
| 400–499 | Điều hướng: Phải thêm thao tác để hoàn thành request |
| 500-599 | Lỗi server: Server không thể đáp ứng một request hợp lệ |

Mỗi HTTP response có một mã response code

Chương 4: Truyền thông với Web server **19**

WEB SERVER

▪ Ở khía cạnh phần cứng, một web server là một máy tính lưu trữ các thành phần của một website. Nó kết nối tới mạng Internet, thường có IP cố định, và thường được truy cập đến thông qua tên miền

▪ Ở khía cạnh phần mềm, một web server là một phần mềm hiểu được các URL (các địa chỉ web) và HTTP (giao thức trình duyệt của bạn sử dụng để xem các trang web), lắng nghe và xử lý các thông điệp được gửi đến từ Web Client (thường là Browser) => Lập trình viên quan tâm đến

Chương 4: Truyền thông với Web server **20**

WEB SERVER

▪ Chức năng chính:

▪ Lưu trữ

▪ Xử lý

▪ Chuyển trang web đến client

▪ Giao tiếp giữa client và server: HTTP

▪ Trang được chuyển: HTTP document, image, css, script Chương 4: Truyền thông với Web server **21**

WEB CLIENT

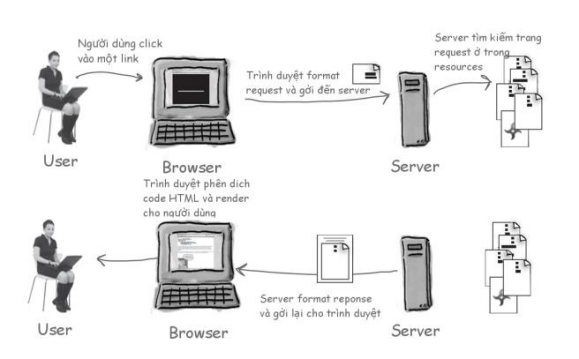
▪ Chức năng chính:

▪ Là phần mềm giúp người dùng giao tiếp với server. ▪ Gửi đi những thông điệp Request dựa vào thao tác của người dung

▪ Tiếp nhận thông điệp Response

▪ Biên dịch HTML và hiển thị (rendering) trang web cho user. Chương 4: Truyền thông với Web server **22**

WEB CLIENT

Chương 4: Truyền thông với Web server **23**

HTTPWEBREQUEST & WEBCLIENT &

HTTPCLIENT

| HTTPWebRequest Tất cả version | WebClient  Tất cả version | HttpClient  Từ .NET  Framework 4.5 |
| --- | --- | --- |
| Class tiêu chuẩn | Bản tóm tắt, đơn giản của HTTPWebRequest | Mạnh mẽ nhất |

Chương 4: Truyền thông với Web server **24**

ĐỂ XÂY DỰNG WEB CLIENT

▪ HttpWebRequest tốt cho kiểm soát

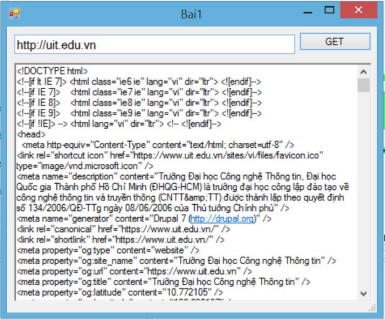
▪ WebClient: đơn giản và ngắn gọn

▪ HttpClient: mạnh mẽ nhất, cho cả hai tính năng trên môi trường .NET 4.5

Chương 4: Truyền thông với Web server **25**

HTTPWEBREQUEST VÍ DỤ

▪ Viết hàm bắt sự kiện nút GET và gọi hàm getHTML, hiển thị thông điệp từ hàm getHTML vào vùng hiển thị (giả sử dùng RichTextBox)

Chương 4: Truyền thông với Web server **26**

HTTPWEBREQUEST VÍ DỤ

private string getHTML(string szUrl)

{

// Khởi tạo 1 HTTPWebRequest cho 1 URL.

HttpWebRequest request = (HttpWebRequest)WebRequest.Create(szUrl); // Nhận thông điệp phản hồi.

HttpWebResponse response = (HttpWebResponse)request.GetResponse(); // Đón nhận stream

Stream dataStream = response.GetResponseStream();

StreamReader reader = new StreamReader(dataStream);

// Đọc nội dung

string responseFromServer = reader.ReadToEnd();

response.Close();

return responseFromServer;

}**27**

HTTPWEBREQUEST

| Phương thức hoặc  thuộc tính | Ý nghĩa |
| --- | --- |
| Accept | Lấy ra hoặc thiết lập giá trị của Accept HTTP header. Kiểu String |
| AllowAutoRedirect | Lấy ra hoặc thiết lập giá trị boolean cho biết có request đi sau các response điều hướng (3xx) hay không |
| ContentLength | Lấy ra hoặc thiết lập Content-length HTTP header |
| ContentType | Lấy ra hoặc thiết lập Content-type HTTP header |
| CookieContainer | Lấy ra hoặc thiết lập các cookie liên kết với request. Ví dụ: CookieContainer.getCookies[“name”].ToString(). |
| Headers | Lấy ra một tập string chứa trong HTTP header. Ví dụ: Headers[“Content-Type”].ToString(). |
| Method | Lấy ra hoặc thiết lập phương thức dành cho request. Có thể thiết lập là GET, HEAD, POST, PUT, DELETE, TRACE, OPTIONS |

Chương 4: Truyền thông với Web server **28**

HTTPWEBREQUEST

| Phương thức hoặc thuộc tính | Ý nghĩa |
| --- | --- |
| Proxy | Lấy ra hoặc thiết lập thông tin Proxy cho request. Trả về WebProxy |
| Referer | Lấy ra hoặc thiết lập giá trị của Referer HTTP header. Trả về String |
| RequestUri | Lấy ra URI gốc của request. Ví dụ: RequestURI.ToString() |
| Timeout | Lấy ra hoặc thiết lập giá trị Timeout. Ví dụ: Timeout=(int) new TimeSpan(0,0,30).TotalMilliseconds |
| TransferEncoding | Lấy ra hoặc thiết lập giá trị giá trị của Transfer-encoding HTTP header. Trả về String |
| UserAgent | Lấy ra hoặc thiết lập giá trị giá trị của User-agent HTTP header. Trả về String |
| GetResponse | Trả về một webResponse từ tài nguyên Internet |

Chương 4: Truyền thông với Web server **29**

CLASS HTTPWEBRESPONSE

| Phương thức hoặc  thuộc tính | Ý nghĩa |
| --- | --- |
| ContentEncoding | Lấy phương pháp dùng để mã hóa nội dung của response. Trả về kiểu String |
| ContentLength | Độ dài của nội dung trả về bởi request, kiểu Long |
| ContentType | Nội dung của response, kiểu String |
| Cookies | Lấy ra hoặc thiết lập các cookie liên kết với request. Ví dụ: Cookies[“name”].ToString() |
| Headers | Lấy ra các header liên kết với response này từ server. Ví dụ: Headers[“Content-Type”].ToString(). |

Chương 4: Truyền thông với Web server **30**

HTTPWEBRESPONSE

| Phương thức hoặc  thuộc tính | Ý nghĩa |
| --- | --- |
| ResponseUri | Lấy ra phần URI của tài nguyên Internet đã được đáp ứng bởi request. Ví dụ: RequestURI.ToString(). |
| Server | Lấy ra tên của server nào gửi response, kiểu String |
| StatusCode | Lấy ra trạng thái của response. Trả về kiểu liệt kê HttpStatusCode |
| GetResponseHeader | Lấy ra nội dung header xác định đã được trả về với response. Kiểu String |
| GetResponseStream | Lấy ra stream dùng để đọc phần thân của response. Kiểu stream |

Chương 4: Truyền thông với Web server **31**

POSTING DATA: VÍ DỤ

Sử dụng Class WebRequest (Lớp trừu tượng) để gửi dữ liệu thông qua HTTP Request:

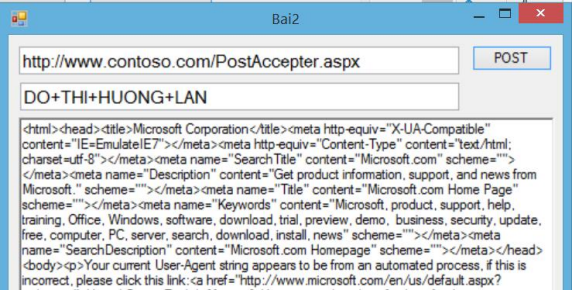
Tham khảo thêm tại:

https://docs.microsoft.com/en

us/dotnet/framework/network-programming/how-to send-data-using-the-webrequest-class

Chương 4: Truyền thông với Web server **32**

POSTING DATA: VÍ DỤ

Chương 4: Truyền thông với Web server **33**

COOKIE

HTTP không duy trì thông tin trạng thái

🡪 khó khăn cho việc phân biệt 2 user truy cập vào server hay 1 user tạo 2 request

🡪 client phải tạo sự khác biệt với client khác

Chương 4: Truyền thông với Web server **34**

COOKIE

▪ Cookie là các file nhỏ lưu trong thư mục

%windows%\cookies, được lưu bằng 2 cách:

▪ Dùng JavaScript document.cookie

▪ Dùng set-cookie header trong các HTTP request

▪ Cookie được lưu giữ trên máy client trong khoảng thời gian xác định

Chương 4: Truyền thông với Web server **35**

COOKIE

▪ Cookie có thể trích xuất được nhờ JavaScript hoặc HTTP response

▪ Cookie được hỗ trợ trong .NET thông qua các đối tượng HttpWebResponse.Cookies và

HttpWebRequest.CookieContainer

▪ Cookie phụ thuộc tên miền (domain)

Cookie lưu cho www.library.com không thể trích xuất bởi www.bookshop.com

Chương 4: Truyền thông với Web server **36**

WEBCLIENT: DOWNLOADING

▪ Class WebClient cung cấp 1 số phương thức để lấy thông tin từ web server:

1. DownloadData(): lấy dữ liệu vào một mảng byte từ URI

2. DownloadFile(): lấy dữ liệu vào một file cục bộ từ URI 3. OpenRead(): mở stream read-only để lấy dữ liệu từ URI

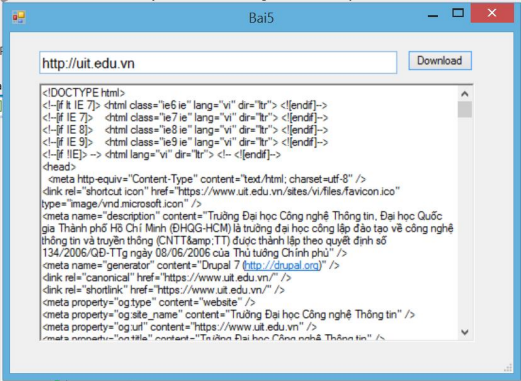
4. Xem them tại https://docs.microsoft.com/en us/dotnet/api/system.net.webclient?view=netframe work-4.8

Chương 4: Truyền thông với Web server **37**

MINH HỌA DOWNLOADDATA

Chương 4: Truyền thông với Web server **38**

MINH HỌA DOWNLOADDATA

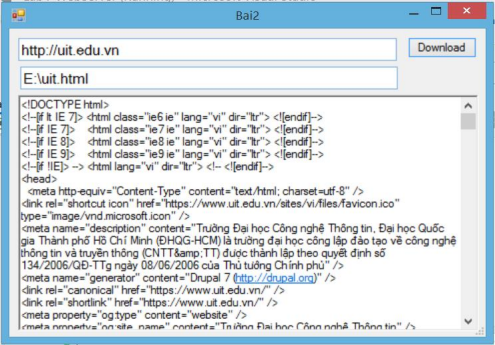
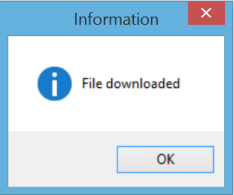
Chương 4: Truyền thông với Web server **39**

MINH HỌA DOWNLOADFILE

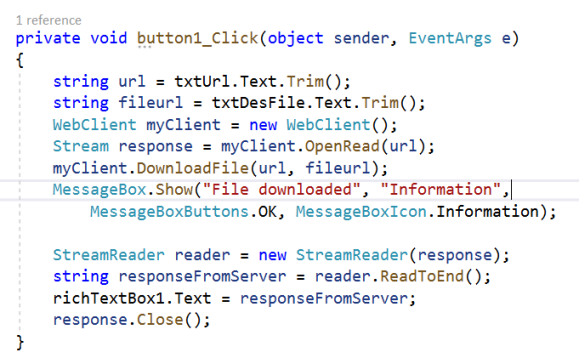
▪ Viết code xử lý sự kiện khi nhấn nút download, với 1 textbox chứa địa chỉ URL cần get nội dung, 1 textbox chứa đường dẫn lưu file chứa nội dung cần get về.

Chương 4: Truyền thông với Web server **40**

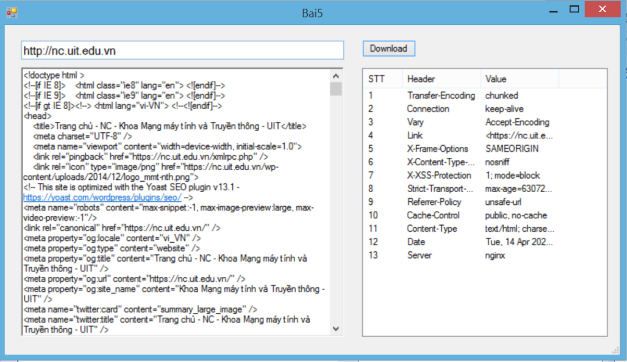
MINH HỌA DOWNLOADFILE

Chương 4: Truyền thông với Web server **41**

MINH HỌA DOWNLOADFILE

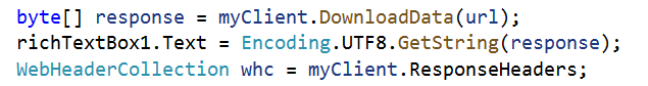
Chương 4: Truyền thông với Web server **42**

WEBCLIENT: PROPERTY

Chương 4: Truyền thông với Web server **43**

WEBCLIENT: PROPERTY

▪ Class WebClient cung cấp thuộc tính ResponseHeaders để lấy thông tin các trường trong HTTP header.

Chương 4: Truyền thông với Web server **44**

WEBCLIENT: UPLOADING

▪ Class WebClient sử dụng 1 số cách để upload thông tin lên web server:

1. OpenWrite(): gửi lên dùng stream

2. UploadData(): gửi lên dùng mảng byte

3. UploadFile(): gửi lên dùng file

4. UploadValues(): gửi một đối tượng

NameValueCollection các data name và value lên web server

https://docs.microsoft.com/en

us/dotnet/api/system.net.webclient?view=netframework 4.8

Chương 4: Truyền thông với Web server **45**

WEBBROWSER CONTROL

▪ Cung cấp giao diện cho phép hiển thị tài liệu HTML, và cung cấp 1 số thuộc tính, phương thức để xây dựng 1 Web Browser cơ bản

https://docs.microsoft.com/en

us/dotnet/api/system.windows.forms.webbrowser?view=ne tframework-4.8

https://docs.microsoft.com/en

us/dotnet/framework/winforms/controls/how-to-add-web browser-capabilities-to-a-windows-forms

application?view=netframework-4.8

Chương 4: Truyền thông với Web server **46**

SYSTEM.NET.HTTPLISTENER

▪ Một trong những phương pháp tốt để hiện thực web server là sử dụng class HttpListener

▪ HttpListener cung cấp Http.sys có rất nhiều chức năng, như chứng thực và mã hóa SSL – nếu tự xây dựng thì tương đối khó khăn

Chương 4: Truyền thông với Web server **47**

HTTPLISTENER

| Phương thức hoặc thuộc tính | Mô tả |
| --- | --- |
| Abort / Close | Hủy bỏ hàng đợi request |
| BeginGetContext | Chờ đợi một client request không đồng bộ. |
| EndGetContext | Quản lý client request . Trả về HttpListenerContext |
| GetContext() | Chờ đợi một client request đồng bộ. Trả về  HttpListenerContext |
| Star | Khởi động web server |
| Stop | Dừng web server |
| AuthenticationSchemes | Thiết lập phương pháp chứng thực giữa server và client (Basic, Digest, NTLM). Trả về AuthenticationScheme |
| IsListening | Xác định xem server có đang chạy hay không |

Chương 4: Truyền thông với Web server **48**