Ví dụ ThreadPool

-Ví dụ về Server sử dụng thread pool

- Quan sát các thread được tạo ra
- Test chương trình Server sử dụng threadpool

CHƯƠNG 6

GiAO TIẾP VỚI WEB SERVER

Nội dung

- Giới thiệu
- HTTP
- Web server
- WebClient
- System.Net.HttpListener
- Mobile Web

Giới thiệu

- Hướng dẫn cách lấy dữ liệu từ Web và sử dụng vào mục đích khác
- Những lý do mà một 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
  - 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

Giới thiệu

- Data mining: tải trang web xuống và khai thác thông tin tự động từ đó
- Để khai thác thông tin có ích từ HTML, cần phải quen thuộc với ngôn ngữ này

213

HTTP

- HTTP hoạt động trên giao thức TCP/IP port 80
- Client mở TCP ở port 80 kết nối đến server
- Client gửi một HTTP request, server hồi đáp với một HTTP response
- Server đóng kết nối TCP

214

HTTP request

- Dạng đơn giản nhất như sau:

  GET /

  <enter><enter>
- Với một số server cần phải xác định DNS name trong lệnh GET
- Request này yêu cầu server trở về trang web mặc định
- Thường có dạng phức tạp hơn như sau:

215

HTTP request

GET / HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg,
application/vnd.ms-powerpoint, application/vnd.ms-excel, application/msword, */*
Accept-Language: en-gb
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT
5.1; .NET CLR 1.0.3705)
Host: 127.0.0.1:90
Connection: Keep-Alive

216

HTTP request

- Thông tin trên cho server biết một số điều về client như: kiểu trình duyệt, phần dữ liệu nào trình duyệt có thể hiển thị

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. Trong ví dụ trên application/msword cho biết trình duyệt hiển thị được tài liệu MS Word
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

217

HTTP request

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. Ví dụ: Host: 127.0.0.1:90 chỉ cho biết client và server nằm cùng một máy tính, chạy tại port 90

218

HTTP request

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 nào 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)

219

HTTP request

- GET và POST là các lệnh HTTP phổ biến
- Ngoài ra còn có HEAD, OPTIONS, PUT, DELETE, TRACE
- Lập trình web thường dùng với mã lệnh HTML có dạng:

```
<form name="myForm"
action="someDynamicPage" method="POST">
```

220

POST request

POST / HTTP/1.1

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

Content-Length: 17

myField=some+text

221

HTTP response

- Khi server nhận được một HTTP request, nó trích xuất trang theo yêu cầu và trả về client cùng với HTTP header. Đó chính là HTTP response
- HTTP response có dạng như sau:

222

HTTP response

HTTP/1.1 200 OK
Server: Microsoft-IIS/5.1
Date: Sun, 05 Jan 2003 20:59:47 GMT
Connection: Keep-Alive
Content-Length: 25
Content-Type: text/html
Set-Cookie:
ASPSESSIONIDQGGQQFCO=MEPLJPHDAGAEHENK
AHIHGHGH;
path=/
Cache-control: private
This is a test html page!

223

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	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

224

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, trong ví dụ trên mã là 200, theo sau là một số văn bản có thể đọc được, đồng nghĩa với nhận thành công

225

Các kiểu MIME

- Multipart Internet mail extensions (MIME)
- Các kiểu MIME mô tả kiểu dữ liệu, giúp cho các máy tính khác hiểu và xử lý phù hợp
- Ví dụ: .JPG được ánh xạ đến image/jpeg, .TXT được ánh xạ đến text/plain
- Để tìm kiểu MIME cho file nào đó, mở registry editor → HKEY_CLASSES_ROOT

226

System.Web

- Cách dùng HTTP phổ biến là khả năng tải nội dung HTML của một trang web lưu vào string
- Ví dụ minh họa:

227

System.Web

```
private string getHTTP(string szURL)
{
    HttpWebRequest  httpRequest;
    HttpWebResponse httpResponse;
    String          bodyText = "";
    Stream          responseStream;
    Byte[] RecvBytes = new Byte[Byte.MaxValue];
    Int32 bytes;
    httpRequest = (HttpWebRequest)
    WebRequest.Create(szURL);
    httpResponse = (HttpWebResponse)
    httpRequest.GetResponse();
    ResponseStream = httpResponse.GetResponseStream();
```

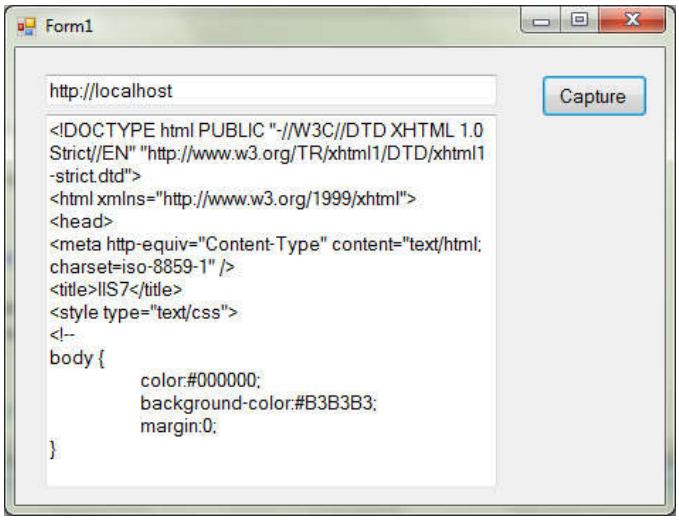
228

System.Web

```
while (true)     {
   bytes = responseStream.Read(RecvBytes,
   0,RecvBytes.Length);
   if (bytes<=0) break;
   bodyText +=
System.Text.Encoding.UTF8.GetString(RecvBytes,
0, bytes);
}
  return bodyText;
}
```

229

System.Web



230

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().

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

Posting data

- Các trang web động chứa các form để đăng nhập, tiêu chuẩn tìm kiếm hoặc dữ liệu khác. Các form này thường được submit thông qua phương thức POST.
- Điều này nảy sinh một số vấn đề vì không thể xác định dữ liệu đã post trong URL
- Các request đến và dữ liệu ra được ánh xạ đến các đối tượng trong .NET

233

Posting data

- Những đối tượng này thường là Request và Response
- Đối tượng Request đóng gói dữ liệu gửi từ trình duyệt đến server. Hai thuộc tính quan trọng của nó gồm: Form và QueryString.
  - Form đọc dữ liệu gửi từ client thông qua phương thức POST
  - QueryString đọc dữ liệu gửi từ client thông qua phương thức GET

234

Posting data

- Đối tượng Response đặt dữ liệu lên HTTP stream để gửi tới client. Một trong những phương thức quan trọng của nó là Write. Write chuyển chuỗi sẽ hiển thị (dạng HTML) cho client
- Một đặc tính khiến ASP.NET mạnh hơn ASP chính là khả năng mô hình hóa các phần tử HTML thành đối tượng, không chỉ đơn thuần là các input stream và output stream

235

Posting data

- Ví dụ: một input box được viết trong ASP.NET dạng <ASP:TEXTBOX id="tbText" runat="server"/> và các thuộc tính của textbox này có thể sửa chữa thông qua việc truy xuất đối tượng tbText
- ASP.NET có hiệu suất tốt hơn ASP vì cách thức biên dịch khi dùng ở lần đầu tiên (in-line) hoặc tiền biên tịch (code-behind)

236

Posting data

- Khi người dùng nhấn vào nút lệnh submit (<input type=”submit”>), trình duyệt đóng gói dữ liệu người dùng nhập vào chứa bên trong các tag <form> và gửi ngược về server như một POST request
- Server phân tích cú pháp POST request nhận được. Server-side script có thể lấy được dữ liệu này bằng cách truy xuất vào Request.Form

Posting data: ví dụ

- Chuẩn bị sẵn script sau:

```
<%@ Page language="c#" Debug="true"%>
<script language="C#" runat="server">
public void Page_Load(Object sender, EventArgs E)
{
if (Request.Form["tbPost"]!=null)
{
 Response.Write(Request.Form["tbPost"].ToString());
}
}
</script>
<form method="post">
<input type="text" name="tbpost">
<input type="submit">
</form>
```

Posting data: ví dụ

- Tạo project mới, có 1 form, 1 button với tên btnCapture. Thêm code xử lý biến cố Click:

```
private void btnCapture_Click(object sender,
System.EventArgs e)
{
    tbPost.Text = HttpUtility.UrlEncode(tbPost.Text);
    tbResult.Text =
getHTTP(tbUrl.Text,"tbPost="+tbPost.Text);
}
```

239

Posting data: ví dụ

```
public string getHTTP(string szURL,string szPost)
{
  HttpWebRequest  httprequest;
  HttpWebResponse httpresponse;
  StreamReader    bodyreader;
  string          bodytext = "";
  Stream          responsestream;
  Stream          requestStream;
  httprequest = (HttpWebRequest) WebRequest.Create(szURL);
  httprequest.Method = "POST";
  httprequest.ContentType =
  "application/x-www-form-urlencoded";
```

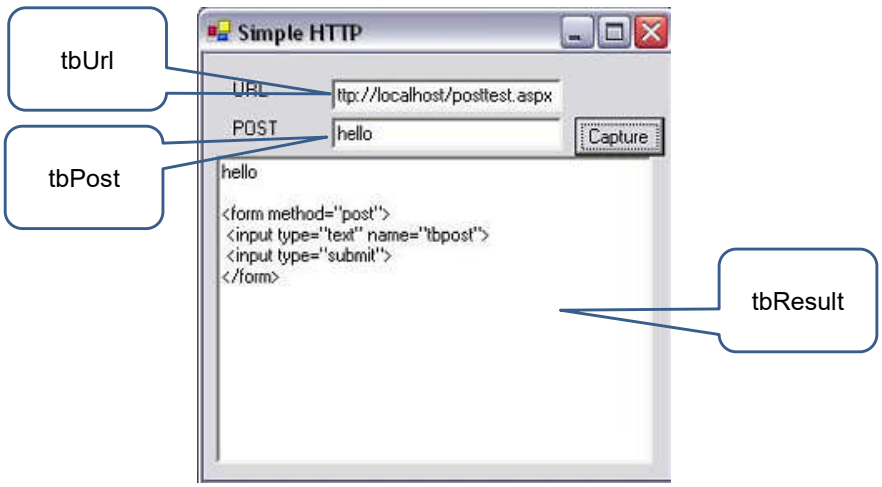
240

Posting data: ví dụ

```
  httprequest.ContentLength = szPost.Length;
  requestStream = httprequest.GetRequestStream();
  requestStream.Write(Encoding.ASCII.GetBytes(szPost),0,
  szPost.Length);
  requestStream.Close();
  httpresponse = (HttpWebResponse)
          httprequest.GetResponse();
  responsestream = httpresponse.GetResponseStream();
  bodyreader = new StreamReader(responsestream);
  bodytext = bodyreader.ReadToEnd();
  return bodytext;
}
```

241

Posting data: kết quả ví dụ



242

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().

243

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

244

Web server

– Web server có thể được xây dựng và cài đặt như một phần của ứng dụng
– Không yêu cầu người dùng cài đặt IIS
– IIS không cài được trên một số HĐH Windows Client

Web server

- "Trái tim" của một HTTP server là một TCP server
- Server phải hỗ trợ multithreaded, vì vậy đầu tiên phải khai báo một mảng các socket:

  private ArrayList alSockets;
- Mỗi HTTP server có một HTTP root – đây là thư mục chứa các trang web

Web server

- Muốn lấy đường dẫn ứng dụng, ta dùng: Application.ExecutablePath → trích xuất được HTTP root
- Lấy các kết nối đến server:

```
alSockets = new ArrayList();
Thread thdListener =
new Thread(new ThreadStart(listenerThread));
thdListener.Start();
```

- Chú ý: hàm listenerThread quản lý các kết nối mới, cấp phát thread cho nó

247

Web server

```
public void listenerThread()
{
  int port =0;
  port = Convert.ToInt16(tbPort.Text);
  TcpListener tcpListener = new TcpListener(port);
  tcpListener.Start();
  while (true)
  {
    Socket handlerSocket = tcpListener.AcceptSocket();
    if (handlerSocket.Connected)
    {
```

248

Web server

```
    lbConnections.Items.Add(
handlerSocket.RemoteEndPoint.ToString() + " connected."    );
    lock(this)
    {
     alSockets.Add(handlerSocket);
     ThreadStart thdstHandler = new
ThreadStart(handlerThread);
     Thread thdHandler = new Thread(thdstHandler);
     thdHandler.Start();
    }
   }
  }
}
```


Web server

- HTTP hoạt động trên port 80, mặc định dành cho IIS, nên ứng dụng khác nếu dùng port này thì sẽ gây ra tranh chấp → ứng dụng hỏng → cần chỉ định port khác
- Thread phải chạy vòng lặp vô tận để đón các kết nối mới, được đặt trong tình trạng blocking với phương thức AcceptSocket().
- Khi socket đã được kết nối, văn bản được viết lên màn hình thì thread mới gọi đến hàm handlerSocket()

Web server

- Lý do phải lock(this) vì handlerSocket trích xuất từ socket bằng cách đọc phần tử cuối cùng trong ArrayList. Trường hợp có 2 kết nối đồng thời đến thì có 2 phần tử được ghi vào ArrayList, do vậy một lần gọi đến handlerSocket sẽ dùng sai socket.
- lock bảo đảm rằng việc sinh ra thread mới không thể xảy ra cùng lúc với quá trình giao tiếp với socket đang mở

251

Web server

- Thread phải được mở trước khi có thể Giao tiếp với client và lấy ở phần tử trên cùng trong danh sách ArrayList. Sau đó tạo stream với client này bằng cách chuyển socket cho hàm khởi tạo đối tượng NetworkStream
- Dùng StreamReader để đọc 1 dòng từ NetworkStream

252

Web server

- Giả sử HTTP request được định dạng đúng, ta có thể trích URL của trang yêu cầu bằng cách tách chuỗi vào 1 mảng
- Chuyển đường dẫn URL thành đường dẫn vật lý trên đĩa cứng cục bộ, gồm 4 bước:
  - Chuyển dấu / thành dấu \
  - Cắt bỏ phần sau dấu ? (dùng để truy vấn)
  - Gắn thêm trang mặc định vào cuối (nếu chưa có)
  - Gắn thêm HTTP root vào đầu URL

253

Web server

- Khi đường dẫn vật lý đã được hình thành, ta có thể thực hiện đọc trên đĩa cứng và gửi đi trên đường truyền mạng (stream)
- Đóng socket
- Một minh họa nhỏ được trình bày trong các slide sau

254

Web server

```
public void handlerThread()
{
  Socket handlerSocket = (Socket) alSockets[alSockets.Count-1];
  String streamData = "";
  String filename = "";
  String[] verbs;
  StreamReader    quickRead;
  NetworkStream networkStream =
  new NetworkStream(handlerSocket);
  quickRead = new StreamReader(networkStream);
  streamData = quickRead.ReadLine();
  verbs = streamData.Split(" ".ToCharArray());
```

255

Web server

```
  filename = verbs[1].Replace("/","\\");
  if (filename.IndexOf("?")!=-1)
  {
    // Trim of anything after a question mark (Querystring)
    filename = filename.Substring(0,filename.IndexOf("?"));
  }

  if (filename.EndsWith("\\"))
  {
    // Add a default page if not specified
    filename+="index.htm";
  }
```

256

Web server

```
 filename = tbPath.Text + filename;
  FileStream  fs = new FileStream(filename,
  FileMode.OpenOrCreate);
  fs.Seek(0, SeekOrigin.Begin);
  byte[] fileContents= new byte[fs.Length];
  fs.Read(fileContents, 0, (int)fs.Length);
  fs.Close();
  // optional: modify fileContents to include HTTP header.
  handlerSocket.Send(fileContents);
  lbConnections.Items.Add(filename);
  handlerSocket.Close();
}
```

257

Web server

- Server không trả về bất kỳ HTTP header nào để cho client biết cách hiển thị thông tin gửi cho nó
- Phần lớn trình duyệt ngày nay có thể xác định cách tốt nhất để hiển thị dữ liệu mà không cần đến Content-Type headers

258

Web server

259

System.Net.HttpWebListener

- Một trong những phương pháp tốt để hiện thực web server là sử dụng class HttpWebListener
- HttpWebListener 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

260

HttpWebListener

Phương thức hoặc thuộc tính	Mô tả
Abort / Close	Hủy bỏ hàng đợi request
AddPrefix	Thêm prefix vào Web listener.
BeginGetRequest	Chờ đợi một client request không đồng bộ. Trả về IasyncResult
EndGetRequest	Quản lý client request . Trả về ListenerWebRequest
GetPrefixes	Trích xuất tất cả prefix đã quản lý. Trả về String[]
GetRequest	Chờ đợi một client request đồng bộ. Trả về ListenerWebRequest
RemoveAll	Gỡ bỏ tất cả prefix
RemovePrefix	Gỡ bỏ một prefix xác định

HttpWebListener

Phương thức hoặc thuộc tính	Mô tả
Star	Khởi động web server
Stop	Dừng web server
AuthenticationScheme	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
Realm string	Nếu phương pháp chứng thực Basic, Digest được chọn thì lấy ra chỉ thị Realm directive. Trả về String

ListenerWebRequest

Phương thức hoặc thuộc tính	Mô tả
Abort / Close	Đóng kết nối client
GetRequestStream	Trích xuất tham chiếu đến stream gửi từ client. Trả về Stream
GetResponse	Trích xuất tham chiếu đến response sẽ gửi đến client. Trả về ListenerWebResponse
Accept	Lấy ra Accept HTTP header gửi trong client request. Trả về String
ClientCertificate	Lấy ra chứng chỉ số gửi với client request. Trả về X509Certificate
ClientCertificateError	Xác định có lỗi nào xảy ra tồn tại trong chứng chỉ. Trả về int32
Connection	Lấy ra Connection HTTP header gửi trong client request. Trả về String

263

ListenerWebRequest

Phương thức hoặc thuộc tính	Mô tả
ContentLength	Lấy ra độ dài của dữ liệu bất kỳ gửi trong client request. Trả về int64
ContentType	Lấy ra ContentType HTTP header gửi trong client request. Trả về String
Expect	Lấy ra Expect HTTP header gửi trong client request. Trả về String
HasEntityBody	Xác định có/không client request chứa Entity body. Trả về Boolean.
Headers	Lấy ra tham chiếu đến tập hợp các HTTP header gửi từ client. Trả về WebHeaderCollection
Host	Lấy ra Host HTTP header gửi trong client request. Trả về String

264

ListenerWebRequest

Phương thức hoặc thuộc tính	Mô tả
Identity	Xác định chứng chỉ nhận dạng trong client request. Trả về Identity
IfModifiedSince	Lấy ra IfModifiedSince header gửi trong client request. Trả về DateTime
KeepAlive Boolean	Xác định có/không client request chứa Connection: Keep-Alive. Trả về Boolean.
LocalEndPoint	Xác định endpoint logic cục bộ của Giao tiếp. Trả về IPEndPoint
Method	Lấy ra phương thức gửi HTTP (GET hoặc POST) gửi trong client request. Trả về String
ProtocolVersion	Xác định phiên bản HTTP dùng bởi client, trả về Version

265

ListenerWebRequest

Phương thức hoặc thuộc tính	Mô tả
RawUri	Lấy ra URI được yêu cầu bởi client. Trả về String
Referer	Lấy ra Referer HTTP header được gửi trong client request. Trả về String
RemoteEndPoint	Xác định endpoint logic ở xa của Giao tiếp. Trả về IPEndPoint
RequestUri	Lấy ra URI được yêu cầu bởi client. Trả về Uri
UserAgent	Lấy ra UserAgent HTTP header được gửi trong client request. Trả về String

266

ListenerWebResponse

Phương thức hoặc thuộc tính	Mô tả
Abort / Close	Hủy kết nối với client
GetResponseStream	Trích xuất tham chiếu đến stream sẽ được trả về từ client. Trả về Stream
ContentLength	Thiết lập độ dài của dữ liệu sẽ gửi cho client. Trả về Int64
ContentType	Thiết lập ContentType HTTP header sẽ gửi cho client. Trả về Int64
Date	Thiết lập Date HTTP header sẽ gửi cho client. Trả về DateTime
EntityDelimitation	Xác định cách thức phân tách nội dung response (ContentLength, Chunked, Raw). Trả về EntityDelimitation.

267

ListenerWebResponse

Phương thức hoặc thuộc tính	Mô tả
Headers	Trích xuất 1 tham chiếu đến HTTP header sẽ gửi về client. Trả về WebHeaderCollection
KeepAlive	Xác định Connection: Keep-Alive được thiết lập trong HTTP header đã trả về cho client. Kiểu boolean
LastModified	Thiết lập LastModified HTTP header sẽ gửi về client. Trả về DateTime
ProtocolVersion	Thiết lập phiên bản giao thức HTTP được dùng Giao tiếp với client. Trả về Version
RawHeaders	Trích xuất 1 tham chiếu đến HTTP header sẽ gửi về client. Trả về ListenerWebRequest
Request	Thiết lập Server HTTP header sẽ gửi về client. Trả về String

268

ListenerWebResponse

Phương thức hoặc thuộc tính	Mô tả
StatusCode	Thiết lập mã trạng thái HTTP sẽ gửi cho client. Trả về httpstatuscode (ví dụ: OK, Moved, NotFound)
StatusDescription	Thiết lập mô tả trạng thái HTTP sẽ gửi cho client. Trả về String

269

WebClient: Downloading

- Class WebClient cung cấp 3 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

270

Minh họa DownloadData

- Tạo project gồm 1 form, 2 textbox với tên tbURL.Text, tbContent; 1button với tên btnGet
- tbContent cho thuộc tính multiline = true
- Xử lý sự kiện Click cho nút lệnh như sau:

Minh họa DownloadData

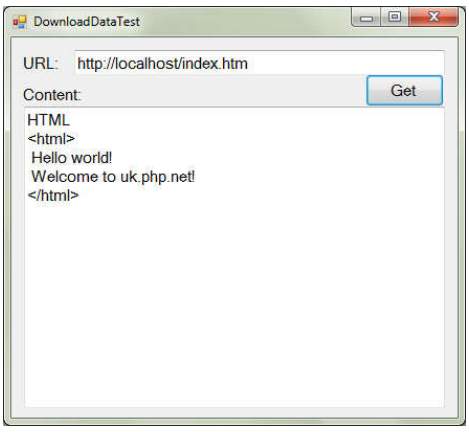
```
private void btnGet_Click(object sender, EventArgs e)
{
    if (tbURL.Text.Trim() == "")
    {
        MessageBox.Show("Please input URL",
"Warning", MessageBoxButtons.OK,
            MessageBoxIcon.Warning);
        return;
    }
    WebClient wc = new WebClient();
```

Minh họa DownloadData

```
    try
    {
        byte[] response = wc.DownloadData(tbURL.Text);
        tbContent.Text =
Encoding.ASCII.GetString(response);
    }
    catch (WebException wex)
    {
        tbContent.Text = wex.Message;
    }
}
```

273

Minh họa DownloadData



274

Minh họa DownloadFile & OpenRead

- Tạo project gồm 1 form, 3 textbox với tên tbURL.Text, tbDesFile.Text, tbContent; 2 button với tên btnDownload, btnGet
- tbContent cho thuộc tính multiline = true
- Xử lý sự kiện Click cho các nút lệnh như sau:

Minh họa DownloadFile & OpenRead

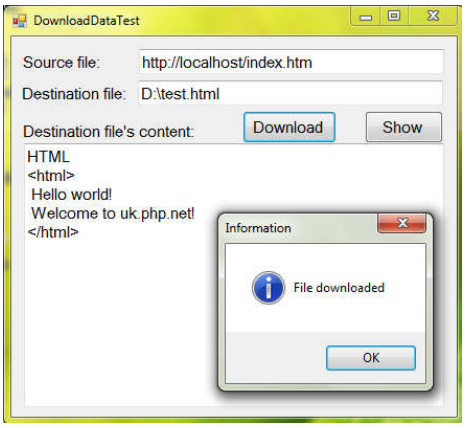
```
private void btnDownload_Click(object sender, EventArgs e)
{
    WebClient wc = new WebClient();
    try {
        wc.DownloadFile(tbURL.Text, tbDesFile.Text);
        tbContent.Text = OpenReader(tbDesFile.Text);
        MessageBox.Show("File downloaded", "Information",
            MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    catch (WebException wex) {
        tbContent.Text = wex.Message;
    }
}
```

Minh họa DownloadFile & OpenRead

```
private string OpenReader(string argv)
{
    string response="";
    Stream strm = wc.OpenRead(argv);
    StreamReader sr = new StreamReader(strm);
    while (sr.Peek() > -1){
        response += sr.ReadLine();
    }
    sr.Close();
    return response;
}
```

277

Minh họa DownloadFile & OpenRead



278

WebClient: property

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

```
public static void Main (string[] argv)
{
  WebClient wc = new WebClient();
  byte[] response = wc.DownloadData(argv[0]);
  WebHeaderCollection whc = wc.ResponseHeaders;
  Console.WriteLine("header count = {0}", whc.Count);
  for (int i = 0; i < whc.Count; i++)  {
    Console.WriteLine(whc.GetKey(i) + " = " + whc.Get(i));
  }
}
```

279

WebClient: Uploading

- Class WebClient sử dụng 4 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

280

OpenWrite

- OpenWrite có thể dùng theo 2 cách:
  1. OpenWrite(string URI): dùng phương thức HTTP POST để gửi dữ liệu từ stream lên web server
  2. OpenWrite(string URI, string method): chỉ định phương thức để gửi dữ liệu lên web server
- Minh họa bằng đoạn code đơn giản như sau:

OpenWrite

```
public static void Main (string[] argv)
 {
  WebClient wc = new WebClient();
  string data = "Data up upload to server";
  Stream strm = wc.OpenWrite(argv[0]);
  StreamWriter sw = new StreamWriter(strm);
  sw.WriteLine(data);
  sw.Close();
  strm.Close();
 }
```

UploadData

- UploadData có thể dùng theo 2 cách:
  1. UploadData(string URI, byte[] array): dùng phương thức HTTP POST để gửi dữ liệu lên web server
  2. UploadData(string URI, string method, byte[] array): chỉ định phương thức để gửi dữ liệu lên web server
- Minh họa bằng đoạn code đơn giản như sau:

UploadData

```
public static void Main (string[] argv)
  {
   WebClient wc = new WebClient();
   string data = "This is the data to post";
   byte[] dataarray =
Encoding.ASCII.GetBytes(data);
   wc.UploadData(argv[0], dataarray);
  }
```

UploadFile

- UploadFile có thể dùng theo 2 cách:
  1. UploadFile(string URI, string *filename*): dùng phương thức HTTP POST để gửi dữ liệu lên web server
  2. UploadFile(string URI, string method, string *filename*): chỉ định phương thức để gửi dữ liệu lên web server

UploadValues

- Khác với các phương thức trên, UploadValues không upload dữ liệu lên web server mà gửi từng cặp name/value cho web server – tương tự như cách truy vấn dữ liệu trong URI
- Phân cách giữa name/value là dấu “&”
- Ví dụ: ?lastname=Blum&firstname=Rich

UploadValues

- UploadValues dùng đối tượng NameValueCollection để quản lý từng cặp name/value
- Ví dụ:

```
NameValueCollection nvc = new
NameValueCollection();
nvc.Add("lastname", "Blum");
nvc.Add("firstname", "Rich");
byte[] response = wc.UploadValues(uri, "POST",
nvc); //có thể dùng GET hoặc POST
```

287

Minh họa UploadValues

```
public static void Main (string[] argv)
{
  WebClient wc = new WebClient();
  string uri = "http://localhost/testform.aspx";
  NameValueCollection nvc = new NameValueCollection();
  nvc.Add("lastname", "Blum");
  nvc.Add("firstname", "Rich");
  byte[] response = wc.UploadValues(uri, nvc);
 Console.WriteLine(Encoding.ASCII.GetString(response));
}
```

288

Bài tập

- Cài đặt các chương trình đã minh họa trong bài giảng của chương

CHƯƠNG 7

GIAO TIẾP VỚI EMAIL SERVER