

CHƯƠNG 5 TRUYỀN THÔNG VỚI EMAIL SERVER

ThS. Trần Bá Nhiệm
Website:
sites.google.com/site/tranbanhiem
Email: tranbanhiem@gmail.com

Nội dung

- Giới thiệu
- Gửi email
- SMTP
- POP3
- System.Web.Mail
- Giao tiếp lập trình ứng dụng mail

Giới thiệu

- Email là phần không thể thiếu trong nhiều lĩnh vực: xã hội, thương mại - kinh tế, quốc phòng,...
- Lập trình để quản lý email là một vấn đề quan trọng và cần thiết
- Trước tiên cần phải hiểu rõ cấu trúc, cách thức gửi của 1 email
- Tiếp cận thông qua lập trình mức socket

29/06/2011

Chương 5: Truyền thông với email
server

3

Gửi email

- Mỗi email đều phải có địa chỉ duy nhất, có dạng: <Username>@<domain name>
- Chú ý: <domain name> phải được nhận diện duy nhất trên hệ thống DNS toàn cầu. <Username> chỉ cần duy nhất trong mail server của người nhận.
- Email không phải được chuyển trực tiếp đến người nhận mà phải đi qua mail server của ISP hoặc của công ty sở hữu domain.

29/06/2011

Chương 5: Truyền thông với email
server

4

Gửi email

- Từ đó mail được chuyển tiếp một lần nữa đến mail server của người nhận sau một khoảng thời gian ngắn
- Gửi mail dùng giao thức SMTP
- Để xác định mail server của người nhận, một mẫu tin MX (Mail Exchange) trong DNS phải có để quản lý

29/06/2011

Chương 5: Truyền thông với email
server

5

SMTP

- SMTP dùng để gửi chứ không nhận mail được
- Mọi mail server phải tuân theo chuẩn SMTP để gửi được đến đích (RFC 821)
- Việc cài đặt một mail server đơn giản để thực hành có thể thông qua hướng dẫn trong bộ cài đặt Windows Server hoặc dùng phần mềm của bên thứ 3 như: AspEmail, Mercury, Pegasus,...

29/06/2011

Chương 5: Truyền thông với email
server

6

Cài đặt SMTP

- SMTP dùng TCP port 25
- Trước khi thực hành nên kiểm tra địa chỉ IP của ISP
- Cách kiểm tra tốt nhất là dùng trình telnet

29/06/2011

Chương 5: Truyền thông với email
server

7

Gửi SMTP mail

- Start → Run: gõ lệnh cmd và click OK
- Gõ lệnh: telnet <server name> 25
- Khi client đã kết nối được với server thì server luôn trả lời với nội dung: 220 <lời chúc mừng> <version-number>
- Khởi động phiên làm việc với các lệnh sau:

29/06/2011

Chương 5: Truyền thông với email
server

8

Gửi SMTP mail

helo

Mail From: <địa chỉ email của người gửi>

Rcpt To: <địa chỉ email của người nhận>

Data

<nội dung của email>

.

Quit

29/06/2011

Chương 5: Truyền thông với email
server

9

Nhận SMTP mail

- Start → Run: gõ lệnh cmd và click OK
- Gõ lệnh: telnet <server name> 110
- Khi client đã kết nối được với server thì server luôn trả lời với nội dung: 220 <lời chúc mừng> <version-number>
- Khởi động phiên làm việc với các lệnh sau:

29/06/2011

Chương 5: Truyền thông với email
server

10

Nhận SMTP mail

User <địa chỉ email của người nhận>

Pass <mật khẩu của người nhận>

Stat //xem trạng thái của hộp thư

List //liệt kê danh sách thư

Retr <số thứ tự thư muốn nhận>

Dele <số thứ tự thư muốn xóa>

Rset //hủy đánh dấu xóa các thư

Quit

29/06/2011

Chương 5: Truyền thông với email
server

11

Lập trình nhận mail bằng .NET

- Khai báo thư viện:
using System.Threading;
using System.Net;
using System.Net.Sockets;
using System.Text;

29/06/2011

Chương 5: Truyền thông với email
server

12

Lập trình nhận mail bằng .NET

- Khởi tạo thread để quản lý ngay khi có thể (tốt nhất nếu viết dạng Form thì nên đặt vào sự kiện Load):

```
private void Form1_Load(object sender,
System.EventArgs e)
{
    Thread thdSMTPServer = new Thread(new
    ThreadStart(serverThread));
    thdSMTPServer.Start();
}
```

29/06/2011

Chương 5: Truyền thông với email
server

13

Lập trình nhận mail bằng .NET

- Thread này cung cấp các chức năng cần thiết để nhận email thông qua SMTP, nó lắng nghe trên port 25 và chờ kết nối đến khi thành công
- Nếu kết nối được thiết lập, thông báo “250 hello<enter>” sẽ được gửi trở lại client
- Đối với SMTP thì chỉ có thể dùng ASCII.GetBytes vì nó là giao thức gửi dựa trên văn bản, dữ liệu nhị phân không được chấp nhận

29/06/2011

Chương 5: Truyền thông với email
server

14

Lập trình nhận mail bằng .NET

- Hàm `socketReadLine` được dùng để lưu trữ dữ liệu đến trong một string cho đến khi gặp ký tự kết thúc
- Hàm `serverThread` thực hiện khởi tạo kết nối TCP, lắng nghe vô thời hạn cho đến khi kết nối bị đóng lại
- Hàm `socketReadLine` trông có vẻ không cần thiết vì lớp `StreamReader` đã có rồi, tuy nhiên hàm này được thiết kế với mục đích riêng – nó có thể phát hiện cả ký tự xuống dòng mới (newline như `'\n'` trong C# hoặc `vbCrLf` trong VB.NET)

29/06/2011

Chương 5: Truyền thông với email
server

15

Lập trình nhận mail bằng .NET

```
public void serverThread()
{
    Byte[] sendBytes;
    TcpListener tcpListener = new TcpListener(25);
    tcpListener.Start();
    while(true) {
        Socket handlerSocket =
        tcpListener.AcceptSocket();
        if (handlerSocket.Connected) {
```

29/06/2011

Chương 5: Truyền thông với email
server

16

Lập trình nhận mail bằng .NET

```

    sendBytes = Encoding.ASCII.GetBytes("250
hello\n");
    handlerSocket.Send(sendBytes,0,
    sendBytes.Length,SocketFlags.None);
    tbStatus.Text +=
socketReadLine(handlerSocket,"\n");
    sendBytes = Encoding.ASCII.GetBytes("250
ok\n");
    handlerSocket.Send(sendBytes,0,

```

29/06/2011

Chương 5: Truyền thông với email
server

17

Lập trình nhận mail bằng .NET

```

    sendBytes.Length,SocketFlags.None);
    tbStatus.Text += socketReadLine(handlerSocket,"\n");
    sendBytes = Encoding.ASCII.GetBytes("250 ok\n");
    handlerSocket.Send(sendBytes,0,
    sendBytes.Length,SocketFlags.None);
    // Wait for enter (rcpt to)
    tbStatus.Text += socketReadLine(handlerSocket,"\n");
    sendBytes = Encoding.ASCII.GetBytes("250 ok\n");
    handlerSocket.Send(sendBytes,0,
    sendBytes.Length,SocketFlags.None);
    // Wait for enter (data)
    tbStatus.Text += socketReadLine(handlerSocket,"\n");

```

29/06/2011

Chương 5: Truyền thông với email
server

18

Lập trình nhận mail bằng .NET

```

sendBytes = Encoding.ASCII.GetBytes("354 proceed\n");
handlerSocket.Send(sendBytes,0,
sendBytes.Length,SocketFlags.None);
// Wait for enter.enter (email body)
tbStatus.Text += socketReadLine(handlerSocket,
"\r\n.\r\n");
// Reply 221 close
sendBytes = Encoding.ASCII.GetBytes("221 close\n");
handlerSocket.Send(sendBytes,0,
sendBytes.Length,SocketFlags.None);
handlerSocket.Close();
} }
}

```

29/06/2011

Chương 5: Truyền thông với email
server

19

Lập trình nhận mail bằng .NET

```

public String socketReadLine(Socket socket,String terminator)
{
    int lastRead=0;
    String streamData = "";
    NetworkStream networkStream = new NetworkStream(socket);
    do {
        lastRead = networkStream.ReadByte();
        if (lastRead==-1) break;
        streamData+=(Convert.ToChar(lastRead));
        if (streamData.EndsWith(terminator)) break;
    } while(true);
    return streamData;
}

```

29/06/2011

Chương 5: Truyền thông với email
server

20

Lập trình nhận mail bằng .NET

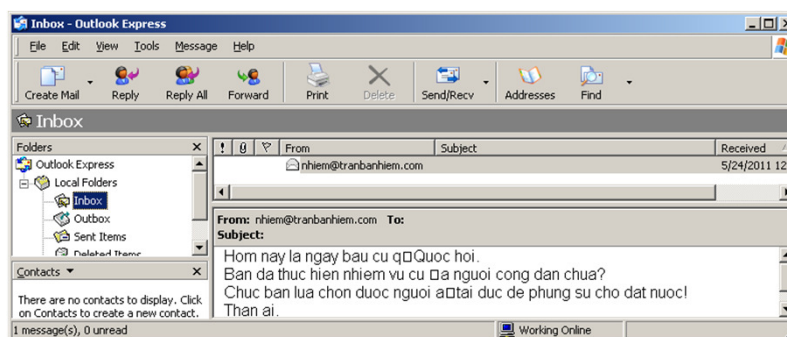
- Để hoàn thành việc kiểm tra chương trình, ta có thể cài đặt một email reader khác như Microsoft Outlook, cấu hình một vài tài khoản email.
- Thực hiện việc gửi email từ tài khoản email đó
- Kiểm tra kết quả bằng chương trình mới viết

29/06/2011

Chương 5: Truyền thông với email server

21

Lập trình nhận mail bằng .NET



29/06/2011

Chương 5: Truyền thông với email server

22

POP3

- Post office protocol 3 (POP3) dùng để nhận chứ không gửi email
- Mọi ISP đều có POP3 server và nhiều công ty cung cấp dịch vụ Web hosting cũng hỗ trợ dịch vụ này
- POP3 được định nghĩa trong RFC 1939, hoạt động trên TCP port 110

29/06/2011

Chương 5: Truyền thông với email
server

23

POP3

- POP3 dùng để lưu trữ email thay mặt cho người dùng, sau đó họ có thể lựa chọn để tải về từ server.
- Một số dịch vụ cung cấp không gian đĩa có giới hạn, do đó cần phải có thao tác xóa email
- Một giải pháp thay thế POP3 là Microsoft Exchange

29/06/2011

Chương 5: Truyền thông với email
server

24

Cài đặt POP3

- Giống như SMTP, POP3 cũng là giao thức dựa trên dòng lệnh
- Mỗi dòng lệnh kết thúc bằng ký tự line-feed (enter).
- Khi server hoạt động bình thường, mỗi dòng sẽ bắt đầu với +OK. Nếu lỗi xảy ra, dòng sẽ bắt đầu với -ERR.

29/06/2011

Chương 5: Truyền thông với email
server

25

Cài đặt POP3

- Để truy xuất được mailbox, client phải chứng thực chính nó với username, password
- Client gửi: USER <username><enter>
- Server trả lời: +OK <welcome><enter>
- Client gửi mật khẩu: USER <password><enter>

29/06/2011

Chương 5: Truyền thông với email
server

26

Cài đặt POP3

- Để xem trạng thái mailbox, client gửi: STAT <enter>
- Server gửi: +OK <số thứ tự> <dung lượng><enter>
- Lấy email chỉ định, clien gửi: RETR <số thứ tự>, trong đó <số thứ tự> thuộc danh sách đã liệt kê trong lệnh STAT
- Server gửi: +OK <some message> <enter> <mail body> <enter>.<enter>

29/06/2011

Chương 5: Truyền thông với email
server

27

Cài đặt POP3

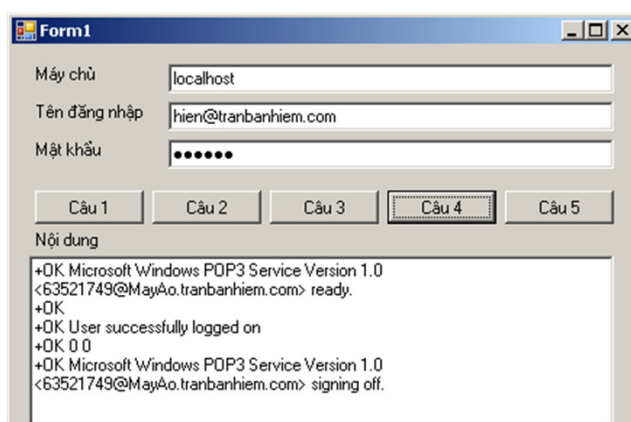
- Để xóa email, client gửi: DELE <số thứ tự>
- Server gửi: +OK <some message> <enter>
- Thực sự đến thời điểm này email vẫn chưa xóa mà chỉ ở tình trạng bị đánh dấu xóa
- Để hủy đánh dấu xóa, clien gửi: RSET
- Để kết thúc phiên làm việc, đóng kết nối TCP, client dùng lệnh: QUIT <enter>

29/06/2011

Chương 5: Truyền thông với email
server

28

Cài đặt POP3



29/06/2011

Chương 5: Truyền thông với email
server

29

Cài đặt POP3

- Thực hiện khai báo các thư viện:
using System.Threading;
using System.Net;
using System.Net.Sockets;
using System.Text;
using System.IO;

29/06/2011

Chương 5: Truyền thông với email
server

30

Cài đặt POP3

- Khai báo các đối tượng để kết nối và nhập/xuất dữ liệu:

```
NetworkStream nStr;
```

```
TcpClient tcp;
```

29/06/2011

Chương 5: Truyền thông với email
server

31

Cài đặt POP3

- Thực hiện kết nối với server:

```
tcp = new TcpClient(txtServer.Text,110);
```

```
nStr = tcp.GetStream();
```

```
StreamReader sr = new StreamReader(nStr);
```

```
txtNoiDung.Text = sr.ReadLine() + "\r\n";
```

```
MessageBox.Show("Connect succeeded!",  
"Information");
```

29/06/2011

Chương 5: Truyền thông với email
server

32

Cài đặt POP3

- Chứng thực:

```
try {
    txtNoiDung.Text += SendPop3("USER " + txtUser.Text
    + "\r\n");
    txtNoiDung.Text += SendPop3("PASS " + txtPass.Text
    + "\r\n");
    MessageBox.Show("Login succeeded!",
    "Information");
}
catch (Exception exp) {
    MessageBox.Show("Error!" + exp.ToString(),
    "Information");
}
```

29/06/2011

Chương 5: Truyền thông với email
server

33

Cài đặt POP3

- Xem trạng thái mailbox:

```
try
{
    txtNoiDung.Text += SendPop3("STAT " + "\r\n");
    WriteFile(txtNoiDung.Text.ToString());
    MessageBox.Show("Succeeded!", "Information");
}
catch (Exception exp)
{
    MessageBox.Show("Error!" + exp.ToString(),
    "Information");
}
```

29/06/2011

Chương 5: Truyền thông với email
server

34

Cài đặt POP3

- Thoát:

```
try
{
    txtNoiDung.Text += SendPop3("QUIT " + "\r\n");
    WriteFile(txtNoiDung.Text.ToString());
    MessageBox.Show("Successted!", "Information");
}
catch (Exception exp)
{
    MessageBox.Show("Error!" + exp.ToString(),
    "Information");
}
```

29/06/2011

Chương 5: Truyền thông với email
server

35

Cài đặt POP3

- Kết thúc phiên làm việc:

```
try
{
    tcp.Close();
    nStr.Close();
    this.Close();
}
catch (Exception exp)
{
    MessageBox.Show("Error!" + exp.ToString(),
    "Information");
}
```

29/06/2011

Chương 5: Truyền thông với email
server

36

Cài đặt POP3

- Nội dung hàm SendPop3:

```
public string SendPop3(string cmd)
{
    StreamReader sr = new StreamReader(nStr);
    byte[] data = Encoding.ASCII.GetBytes(
        cmd.ToCharArray());
    nStr.Write(data, 0, data.Length);
    return sr.ReadLine() + "\r\n";
}
```

29/06/2011

Chương 5: Truyền thông với email
server

37

Cài đặt POP3

- Nội dung hàm WriteToLogs:

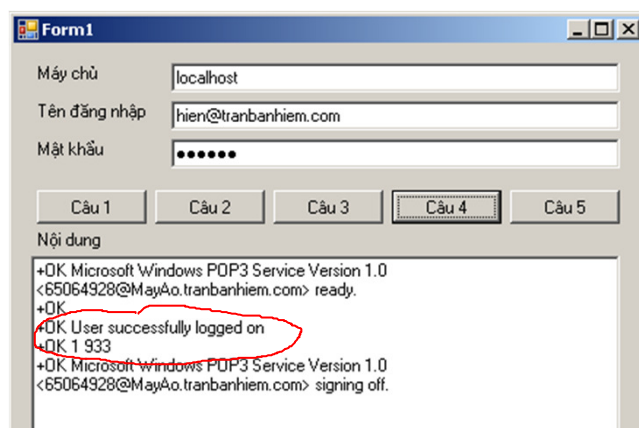
```
public void WriteToLogs(string msg, FileStream
file)
{
    byte[] bData = Encoding.ASCII.GetBytes(
        msg.ToCharArray());
    file.Write(bData, 0, bData.Length);
    file.Flush();
}
```

29/06/2011

Chương 5: Truyền thông với email
server

38

Cài đặt POP3

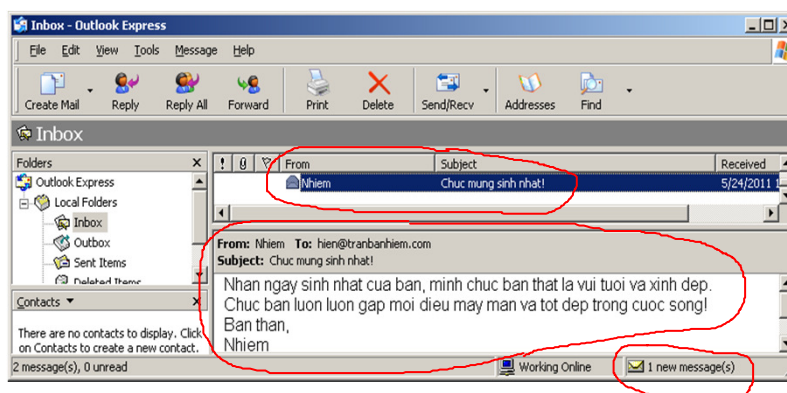


29/06/2011

Chương 5: Truyền thông với email server

39

Cài đặt POP3



29/06/2011

Chương 5: Truyền thông với email server

40

Cài đặt POP3: xóa email

```
TcpClient client = new TcpClient(tbServer.Text, 110);
NetworkStream networkStream = client.GetStream();
StreamReader reader = new
StreamReader(networkStream);
string welcomeMessage = reader.ReadLine();
lstLogs.Items.Add(welcomeMessage);
RunCmd("USER " + tbUser.Text + NEW_LINE,
networkStream);
RunCmd("PASS " + tbPass.Text + NEW_LINE,
networkStream);
RunCmd("DELE 1" + NEW_LINE, networkStream);
RunCmd("QUIT\r\n", networkStream);
```

29/06/2011

Chương 5: Truyền thông với email
server

41

Cài đặt POP3: xóa email

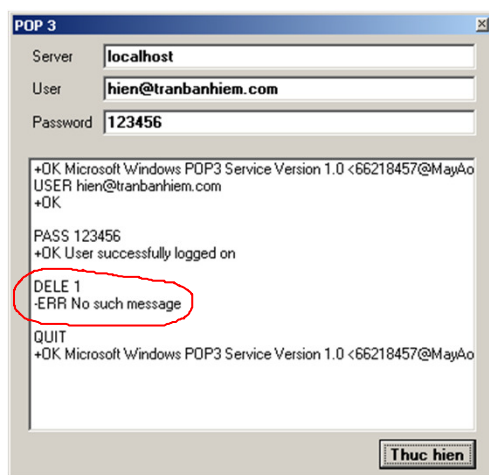
```
private void RunCmd(string cmd, NetworkStream
networkStream)
{
    byte[] bData =
Encoding.ASCII.GetBytes(cmd.ToCharArray());
    networkStream.Write(bData, 0, bData.Length);
    StreamReader reader = new
StreamReader(networkStream);
    string retValue = reader.ReadLine();
    WriteToLogs(retValue + NEW_LINE, file);
    lstLogs.Items.Add(cmd.Replace(NEW_LINE, ""));
    lstLogs.Items.Add(retValue);
    lstLogs.Items.Add("");
}
```

29/06/2011

Chương 5: Truyền thông với email
server

42

Cài đặt POP3: xóa email



29/06/2011

Chương 5: Truyền thông với email
server

43

Bài tập ứng dụng: SPAM filter

- Mục tiêu: chương trình sẽ quét trong mailbox, kiểm email nào chứa văn bản đã chỉ định để thực hiện xóa nó
- Nội dung phần kết nối và chứng thực tương tự như đã viết ở trên. Trình bày ở slide sau là những kỹ thuật mới về cách quét tất cả các mail trong mailbox, tìm kiếm theo nội dung (giả sử “freemoney”)

29/06/2011

Chương 5: Truyền thông với email
server

44

Bài tập ứng dụng: SPAM filter

```
TcpClient clientSocket = new TcpClient( tbServer.Text,
110);
NetworkStream NetStrm = clientSocket.GetStream();
StreamReader RdStrm= new StreamReader(NetStrm);
string Data = sendPOP3("STAT\r\n",NetStrm);
string[] BreakDown = Data.Split(" ".ToCharArray());
int messageCount = Convert.ToInt16(BreakDown[1]);
for (int i=1;i<= messageCount;i++)
{
    StringBuilder message = new StringBuilder("");
    Data = "RETR " + Convert.ToString(i) + "\r\n";
    byte[] szData=
    System.Text.Encoding.ASCII.GetBytes(Data.ToCharArray());
```

29/06/2011

Chương 5: Truyền thông với email
server

45

Bài tập ứng dụng: SPAM filter

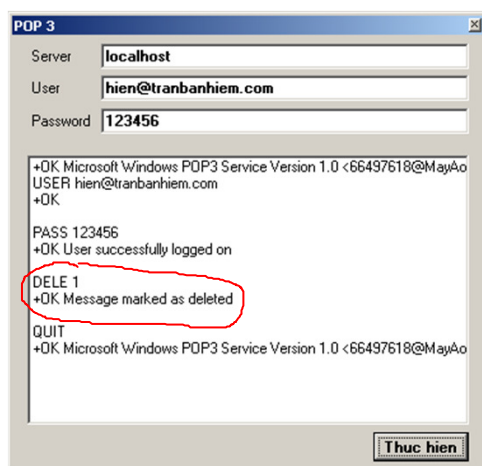
```
NetStrm.Write(szData,0,szData.Length);
string szTemp = RdStrm.ReadLine();
while(szTemp!=".") {
    message.Append(szTemp);
    tbStatus.Text += szTemp+"\r\n";
    szTemp = RdStrm.ReadLine();
}
if (message.ToString().IndexOf("free money")>0) {
    sendPOP3("DELE " + Convert.ToString(i) +
    "\r\n",NetStrm);
}
}
```

29/06/2011

Chương 5: Truyền thông với email
server

46

Cài đặt POP3: xóa email



29/06/2011

Chương 5: Truyền thông với email
server

47

System.Web.Mail

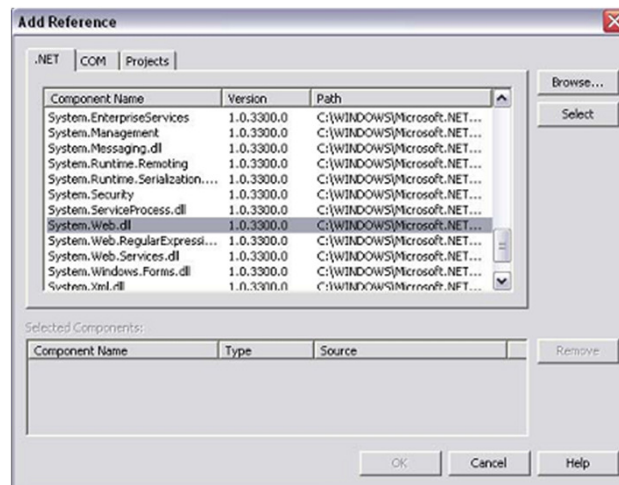
- System.Web.Mail là công cụ có sẵn trong Windows là được dùng cho việc gửi email.
- Chúng đơn giản hơn SMTP, đặc biệt trong việc đính kèm file và văn bản rich-text
- Ta phải thêm reference đến System.Web.dll trước khi có thể dùng namespace System.Web.Mail

29/06/2011

Chương 5: Truyền thông với email
server

48

System.Web.Mail



29/06/2011

Chương 5: Truyền thông với email server

49

System.Web.Mail

```
using System.Web.Mail;
private void btnSend_Click(object sender,
System.EventArgs e)
{
    MailMessage email = new MailMessage();
    email.From = tbFrom.Text;
    email.To = tbTo.Text;
    email.Subject = "email from .NET";
    email.Body = tbMessage.Text;
    SmtpMail.SmtpServer = tbServer.Text;
    SmtpMail.Send(email);
}
```

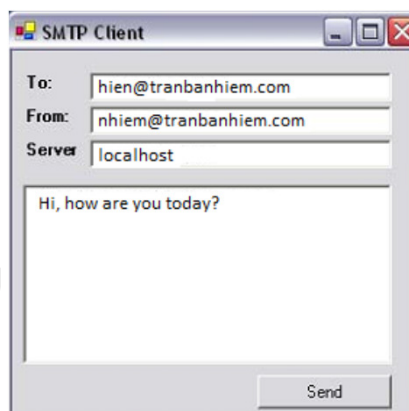
29/06/2011

Chương 5: Truyền thông với email server

50

System.Web.Mail

- Đoạn chương trình trên đơn giản thiết lập các thuộc tính cho đối tượng MailMessage và truyền cho đối tượng SmtMail để gửi email đi



29/06/2011

Chương 5: Truyền thông với email server

51

Attachments (Đính kèm)

- Việc đính kèm thêm các đối tượng file khác là điều hiển nhiên phải hỗ trợ trong các email hiện đại
- Để thực hiện việc này chúng ta có thể làm như sau:

29/06/2011

Chương 5: Truyền thông với email server

52

Attachments (Đính kèm)

```
private void btnSend_Click(object sender,
System.EventArgs e)
{
    MailMessage email = new MailMessage();
    MailAttachment fileAttachment=new
    MailAttachment(tbAttachment.Text);
    email.Priority = MailPriority.High;
    email.BodyFormat = MailFormat.Html;
    email.From = tbFrom.Text;
```

29/06/2011

Chương 5: Truyền thông với email
server

53

Attachments (Đính kèm)

```
email.To = tbTo.Text;
email.Subject = "email from .NET";
email.Body = tbMessage.Text;
email.Attachments.Add(fileAttachment);
SmtpMail.SmtpServer = tbServer.Text;
SmtpMail.Send(email);
}
```

29/06/2011

Chương 5: Truyền thông với email
server

54

Attachments (Đính kèm)

- Đối với các file hình ảnh thì cần thêm một chút xử lý
- Trên web, muốn hiển thị hình ảnh ta dùng: ``, nhưng không thể làm điều đó trên email
- Cách xử lý như sau:
 - Dùng `` trong body email
 - Gọi `attachInlineFile("c:\picture.jpg", "", "picture1")`
 - Gọi hàm Send email

29/06/2011

Chương 5: Truyền thông với email
server

55

Giao tiếp lập trình ứng dụng mail

- Microsoft Outlook có cung cấp 1 interface để các ứng dụng truy xuất email lưu trữ bên trong, được gọi là Mail Application Programming Interface (MAPI), là thư viện kiểu COM (Component Object Model)
- Chúng ta vẫn có thể truy xuất được COM này từ môi trường .NET

29/06/2011

Chương 5: Truyền thông với email
server

56

Giao tiếp lập trình ứng dụng mail

- Để thêm COM trên vào, ta chọn: Project → Add Reference: chọn COM và tìm đến Microsoft Outlook 10.0 Object Library → Select
- Xử lý:
 ListViewItem liEmail;
 Microsoft.Office.Interop.Outlook.Application App;
 Microsoft.Office.Interop.Outlook.MailItem Msg;
 Microsoft.Office.Interop.Outlook.NameSpace NS;
 Microsoft.Office.Interop.Outlook.MAPIFolder Inbox;
 Microsoft.Office.Interop.Outlook.Items Items;
 App = new
 Microsoft.Office.Interop.Outlook.Application();

29/06/2011

Chương 5: Truyền thông với email
server

57

Giao tiếp lập trình ứng dụng mail

```

NS = App.GetNamespace("mapi");
Inbox = NS.GetDefaultFolder(
Microsoft.Office.Interop.Outlook.OlDefaultFolders.olFolderInbox);
Items = Inbox.Items;
for (int i = 1; i < Items.Count; i++)
{
    Msg =
    (Microsoft.Office.Interop.Outlook.MailItem)Items[i];
    liEmail = lvOutlook.Items.Add(Msg.SenderName);
    liEmail.SubItems.Add(Msg.Subject);
}

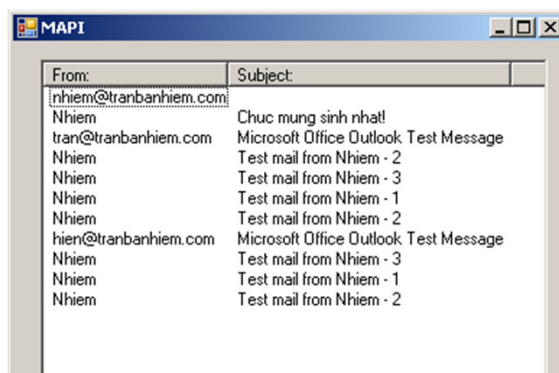
```

29/06/2011

Chương 5: Truyền thông với email
server

58

Giao tiếp lập trình ứng dụng mail



29/06/2011

Chương 5: Truyền thông với email
server

59

Truy xuất sổ địa chỉ

- MAPI có thể được dùng để truy cập hầu hết các đặc tính của Microsoft Outlook
- Sổ địa chỉ được truy xuất thông qua AddressLists collection trong MAPI namespace. Mỗi phần tử trong đó chứa một AddressEntries collection. Và tiếp theo mỗi phần tử trong này chứa các thuộc tính Name và Address
- Code minh họa như sau:

29/06/2011

Chương 5: Truyền thông với email
server

60

Truy xuất sổ địa chỉ

```

ListViewItem liEmail;
Microsoft.Office.Interop.Outlook.Application App =
null;
Microsoft.Office.Interop.Outlook.NameSpace NS =
null;
try
{
    App = new
Microsoft.Office.Interop.Outlook.Application();
    NS = App.GetNamespace("mapi");
}

```

29/06/2011

Chương 5: Truyền thông với email
server

61

Truy xuất sổ địa chỉ

```

catch
{
    MessageBox.Show("Kết nối với Outlook bị lỗi",
"Thông báo lỗi",
    MessageBoxButtons.OK,
    MessageBoxIcon.Error);
}
Microsoft.Office.Interop.Outlook.AddressList
CurrentList;
Microsoft.Office.Interop.Outlook.AddressEntry
CurrentEntry;

```

29/06/2011

Chương 5: Truyền thông với email
server

62

Truy xuất sổ địa chỉ

```

for (int ListsIndexer = 1; ListsIndexer <=
NS.AddressLists.Count; ListsIndexer++)
{
    CurrentList = NS.AddressLists[ListsIndexer];
    for (int EntriesIndexer = 1; EntriesIndexer <=
CurrentList.AddressEntries.Count; EntriesIndexer++)
    {
        CurrentEntry =
CurrentList.AddressEntries[EntriesIndexer];
        liEmail = lvOutlook.Items.Add(CurrentEntry.Name);
        liEmail.SubItems.Add(CurrentEntry.Address);
    }
}

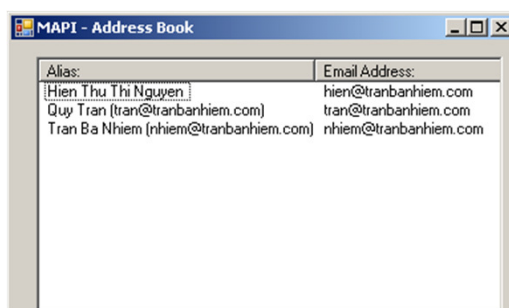
```

29/06/2011

Chương 5: Truyền thông với email
server

63

Truy xuất sổ địa chỉ



29/06/2011

Chương 5: Truyền thông với email
server

64

IMAP

- Internet message access protocol (IMAP) chạy trên port 143 và được định nghĩa trong RFC 1730
- Mặc dù SMTP và POP3 là các chuẩn thực tế của truyền thông email.
- IMAP là công nghệ cạnh tranh với nhiều đặc điểm vượt trội, tuy nhiên còn vài lý do nên ít phổ biến

29/06/2011

Chương 5: Truyền thông với email
server

65

IMAP

- Email lưu trữ trong IMAP server có thể được đánh dấu như đã trả lời, đính flag, xóa, xem,...
- Các flag giúp một tài khoản IMAP được dùng trên nhiều client. Nếu 1 tài khoản POP3 truy cập trên nhiều client sẽ khó theo dõi tình trạng
- Giao thức tương tự như POP3 nhưng phức tạp và có cú pháp linh hoạt hơn

29/06/2011

Chương 5: Truyền thông với email
server

66

IMAP

- Các tiến trình làm việc cũng khá tương tự với POP3 gồm:
 - Chứng thực: clien gửi login <username> <password>. Nếu đúng thì server trả lời: OK LOGIN completed
 - Lấy thông tin về mailbox: clien gửi select inbox. Server trả lời: * <số lượng mail> EXISTS
 - Lấy một email: clien gửi fetch <số thứ tự>

29/06/2011

Chương 5: Truyền thông với email
server

67

IMAP

- Server trả lời với nội dung email theo chuẩn RFC 822 và thông báo OK FETCH completed
- Xóa một email: clien gửi store <số thứ tự> +flags \deleted

29/06/2011

Chương 5: Truyền thông với email
server

68

Network news transfer protocol

- Network news transfer protocol (NNT) chạy trên port 119 và được định nghĩa trong RFC 977
- Giao thức này dùng để quản lý mailing list và hiện nay trở nên lạc hậu
- 2 tác vụ chính:
 - Đọc tin mới
 - Tạo tin mới

29/06/2011

Chương 5: Truyền thông với email server

69

Bài tập

- Cài đặt các chương trình đã minh họa trong bài giảng của chương bằng ngôn ngữ C# hoặc VB.NET

29/06/2011

Chương 5: Truyền thông với email server

70