Chương 1

GIỚI THIỆU ASP

Mục tiêu

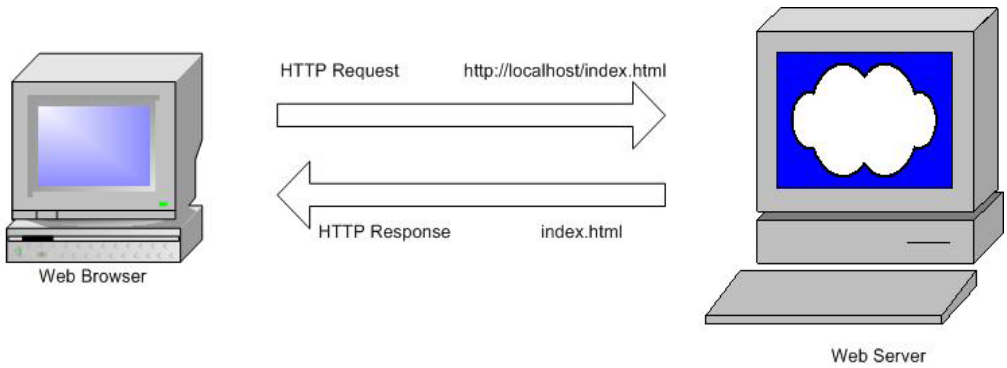
- Giới thiệu ngôn ngữ ASP
- Cài đặt và chạy ứng dụng ASP trên server IIS
- Các cú pháp căn bản VBScript
- Các đối tượng có sẵn
- Thao tác với Database trong ASP

1.1 Giới thiệu ngôn ngữ lập trình web động ASP

Các website thuở ban đầu chỉ bao gồm các trang web tĩnh dưới dạng các file HTML, tất cả những gì cần hiển thị trên trang web thì người thiết kế phải tạo sẵn trên trang đó. Các trang web tĩnh có đuôi là .htm hoặc .html
Chẳng hạn muốn tạo một trang web có hiển thị chữ "Hello" với màu chữ đỏ người ta viết file index.html với nội dung như sau:

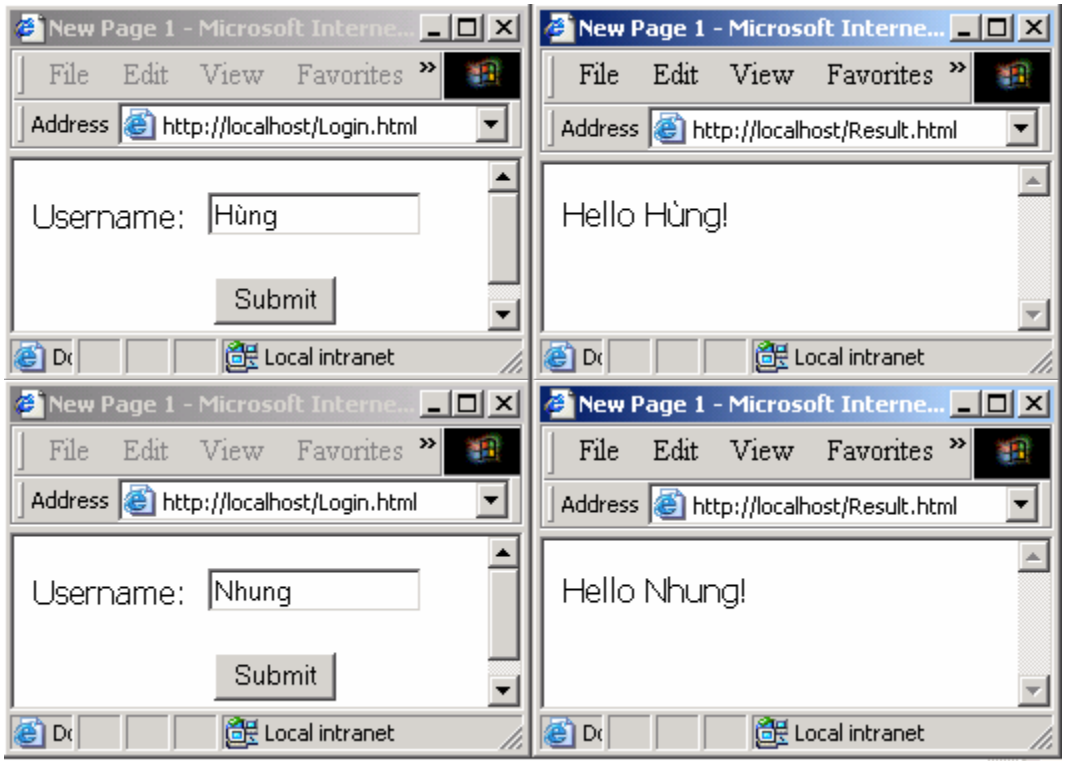
```
<html>
<head>
<title>index</title>
</head>
<body>
<p><font color="red">Hello</font></p>
</body>
</html>
```

Trang web sau đó sẽ được lưu trên Web Server. Khi người dùng muốn xem trang web này họ sẽ dùng trình duyệt gửi một yêu cầu đến server bằng cách gõ vào địa chỉ URL ví dụ : http://localhost/index.html
Lúc này Web Server nhận được yêu cầu sẽ tìm trong kho dữ liệu của nó trang web index.html tương ứng rồi gửi về cho client, sau đó trang web này sẽ được hiển thị ra bởi trình duyệt.
Đó là cách hoạt động của web tĩnh.



Hình 1.1 Cách hoạt động của web tĩnh

Trang web tĩnh tuy rất tiện lợi nhưng không thể đáp ứng được mọi nhu cầu của ứng dụng web, đặc biệt là những yêu cầu tương tác giữa client và web server. Có nhiều tình huống mà nội dung trang web không phải lúc nào cũng có thể soạn thảo và lưu trữ sẵn được mà đôi khi nó cần được sinh ra một cách tự động tùy thuộc vào ngữ cảnh; hoặc có những xử lý phức tạp hơn việc server chỉ đơn giản trả về trang html khi nhận được yêu cầu từ người dùng, ví dụ như phải thu thập thông tin mà người dùng gửi lên qua URL hay form, hoặc truy cập dữ liệu trong database. Lấy ví dụ nếu chúng ta muốn xây dựng một trang web Login.htm yêu cầu người sử dụng nhập tên username, sau khi submit web server sẽ gửi về người dùng trang web Result.html có nội dung : Welcome username!
Dễ dàng thấy rằng trang Result.htm không thể soạn thảo sẵn được vì ứng với mỗi username mà người dùng nhập vào, trang này có nội dung khác nhau.



Hình 1.2 Trang Result.html có nội dung khác nhau tùy vào tương tác giữa client và webserver. Nó không thể soạn thảo sẵn!

Nghĩa là các trang web tĩnh không có khả năng tương tác với người dùng.
Trong thực tế có rất nhiều trường hợp chúng ta thường gặp trong thế giới web đòi hỏi sự tương tác mà web tĩnh không thể giải quyết được (chat, forums, web mail, trang tin tức, giỏ hàng, thông tin thời tiết từng ngày, tỷ giá ngoại tệ hàng ngày)

Để giải quyết vấn đề này người ta sử dụng các ngôn ngữ lập trình web để hỗ

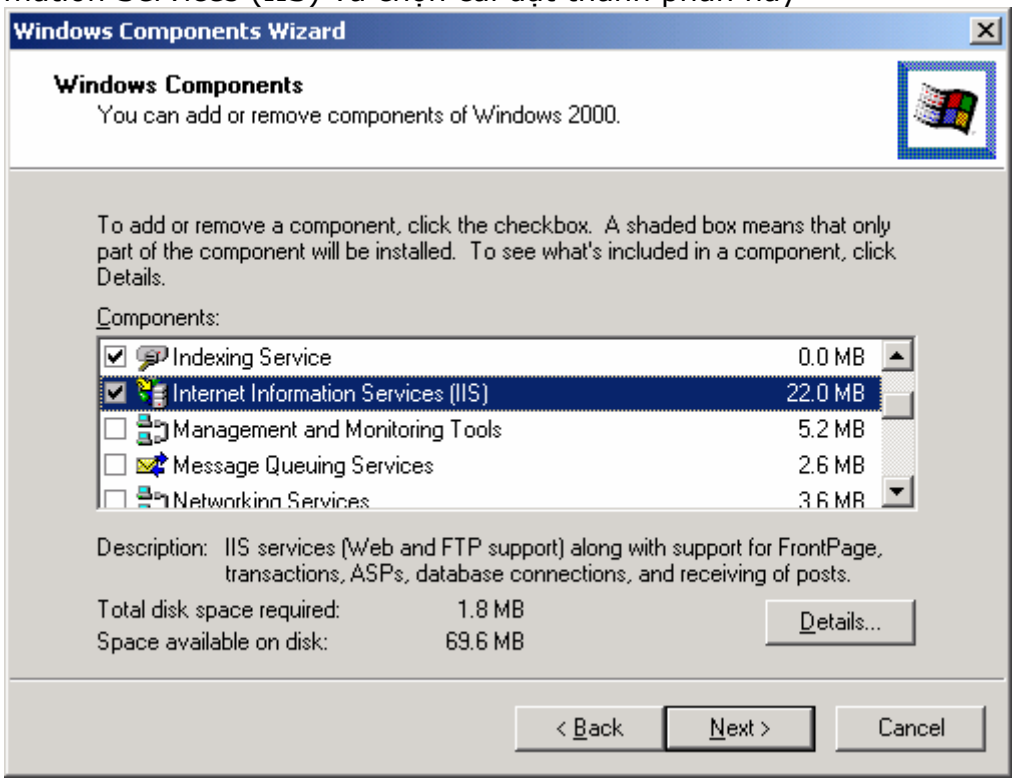
trợ sự tương tác giữa client và server. Chúng là những file có chứa các mã lập trình, có thể tạo ra các trang web động, cho phép trả về cho client trang web có nội dung có thể thay đổi một cách linh động ứng với những ngữ cảnh cụ thể, thu thập và phản hồi với thông tin mà người dùng gửi lên server (thông qua form hay URL), truy cập dữ liệu trong database...

Một số ngôn ngữ lập trình web động phổ biến gồm ASP, PHP, Java, .net ...
ASP (Active Server Pages) là ngôn ngữ lập trình web được viết bởi hãng Microsoft, rất phổ biến trên hệ điều hành Microsoft Windows. Các trang web viết bằng ngôn ngữ này có phần mở rộng là .asp (ví dụ HelloWorld.asp) thay vì .htm hay .html. Nội dung file ASP về cơ bản rất giống file Html bình thường, nó bao gồm các cú pháp html trộn lẫn các mã lập trình ASP (còn gọi là các script, được viết bằng VBScript hay JavaScript). Các Script trong ASP thực thi trên server.
Có thể nói trang ASP là sự kết hợp các thẻ html, các script và các ActiveX Component. Script có thể trộn lẫn giữa các thẻ html và nằm trong cặp dấu <% %>

1.2 Web Server IIS

Thông thường người ta dùng ASP với Web Server có tên là Internet Information Services (IIS) của Microsoft. Đây là thành phần có sẵn trong hệ điều hành Windows 2000 hoặc XP.

Nếu máy tính chưa cài đặt thì chúng ta có thể vào Control Panel => Add/remove programs=> Add/remove Windows Components=>Internet Information Services (IIS) và chọn cài đặt thành phần này



1.3 Cài đặt và chạy ứng dụng ASP đầu tiên

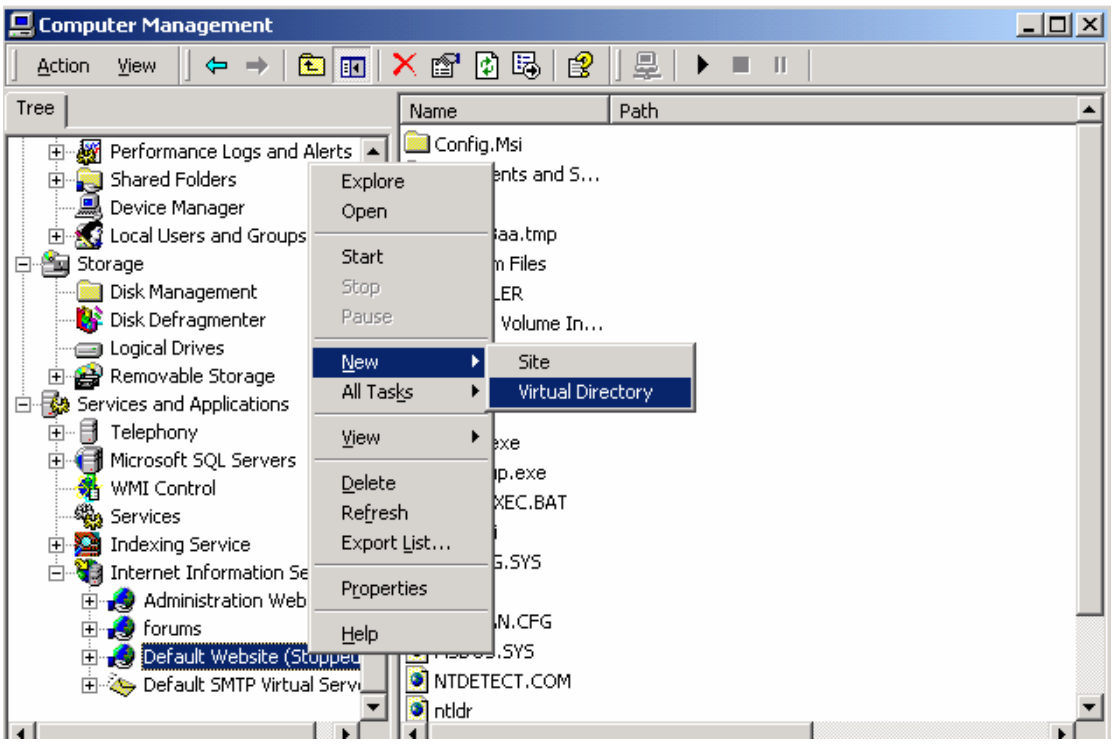
Để bắt đầu chạy một website viết bằng ngôn ngữ ASP đầu tiên chúng ta thực hiện các bước sau:

- Cài đặt web server IIS (ở phần trên) và start IIS
- Cấu hình cho website bằng cách tạo Virtual Directory trên Web Server
- Viết các file ASP và save vào thư mục đã được cấu hình cho website trên server
- Dùng trình duyệt (như Internet Explorer) trên client yêu cầu file ASP và hiển thị kết quả trả về.

1.3.1 Cấu hình cho Website trên IIS

Sau khi start IIS mặc định web server sẽ phục vụ ở địa chỉ http://localhost (địa chỉ trên máy local, cũng giống như một địa chỉ website kiểu như http://www.yahoo.com trên Internet)
Chúng ta tạo một thư mục ảo (Virtual Directory) trên web server để chứa ứng dụng web, ví dụ http://localhost/test
ở đây "test" còn được gọi là Alias của Virtual Directory này. Vậy để lưu trữ các trang ASP trên server trước hết ta sẽ tạo một Virtual Directory với một Alias và thư mục tương ứng rồi upload các file ASP vào thư mục này, sau đó truy cập các trang ASP này thông qua địa chỉ http://localhost/Alias

Cách tạo một Virtual Directory trong IIS:
Vào Web Server từ Control Panel=> Administrative Tools=>Internet Services Manager (hoặc Computer Management)=> Default Website (nếu thấy nó đang stop thì start nó lên) => New=> Virtual Directory (làm theo wizard, chọn các tham số Alias: tên Virtual Directory của mình ví dụ "test", Directory: thư mục chứa Website ví dụ "C:\Web")



Hình 1.4 Tạo Virtual Directory trên IIS

Sau khi kết thúc wizard này chúng ta đã có một Virtual Directory sẵn sàng trên web server. Hãy save các trang asp vào thư mục "c:\Web". Địa chỉ truy cập vào website trong trường hợp này sẽ là: http://localhost/test/

Một cách khác cũng tương tự và dễ thao tác hơn là nhấn chuột phải vào thư mục C:\web, chọn Properties => Web sharing => Share this folder=> Add Alias.

1.3.2 Viết các file ASP

Script được viết trong cặp thẻ <% %>, bắt đầu bằng thẻ mở <% và kết thúc bằng thẻ đóng %>
Chúng ta có thể soạn trang ASP bằng bất cứ chương trình soạn thảo nào như notepad, Frontpage, Dreamweaver...
Ví dụ, tạo 1 file Hello.asp để hiển thị lời chào Hello ra màn hình, save vào thư mục "c:\Web"

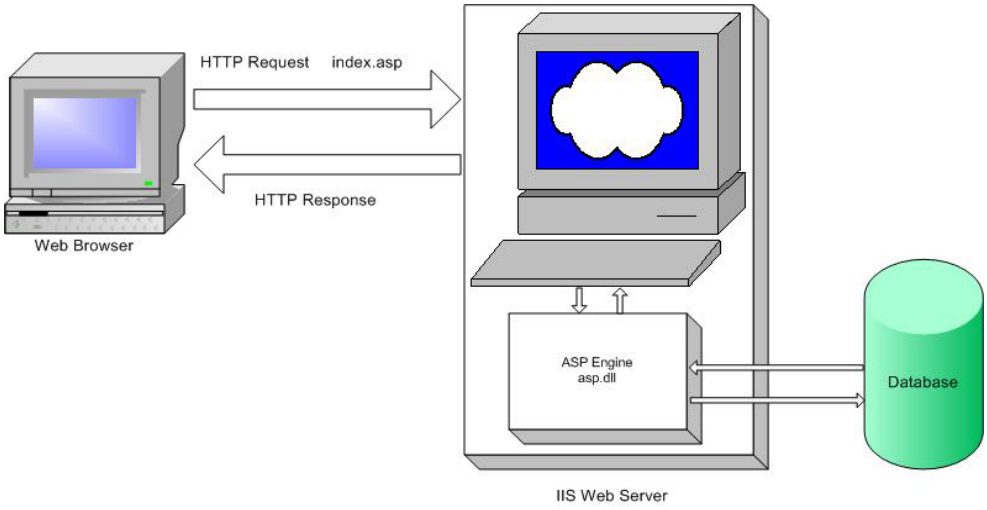
```
<html>
<head>
<title>New Page 1</title>
</head>
<body>
<%
response.write "Hello!"        'Hiển thị lời chào Hello
%>
</body>
```

</html>
Câu lệnh response.write sẽ cho phép hiển thị một chuỗi ra trang web.
Chú thích trong lập trình ASP được viết sau dấu nháy đơn '
Mã lập trình ASP <%response.write "Hello!" %> được viết trộn lẫn giữa các thẻ HTML.

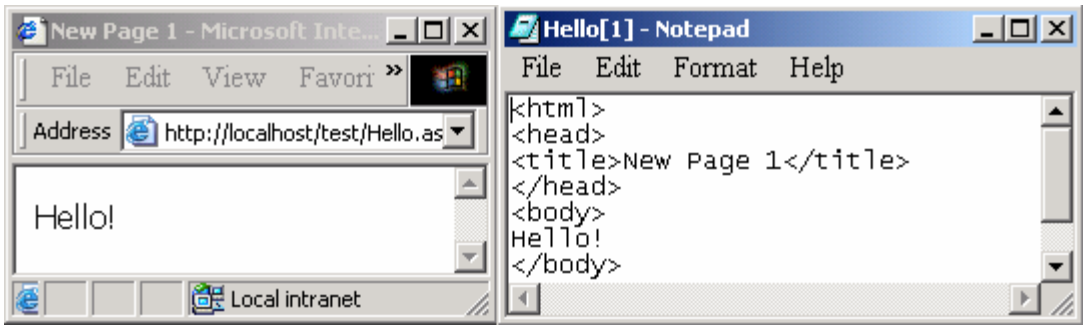
1.3.3 Dùng trình duyệt truy cập website

Mở trình duyệt (ví dụ Internet Explorer), trên thanh địa chỉ gõ địa chỉ sau đây để truy cập vào trang Asp ta đã tạo ra:
http://localhost/test/Hello.asp
Lưu ý là trang asp phải chạy trên web server chứ không thể open trực tiếp với browser như các trang html.

Webserver xử lý như thế nào khi người dùng yêu cầu một trang ASP: Không giống như html, khi người dùng yêu cầu 1 trang html, web server sẽ tìm trong kho dữ liệu và trả về file html đó để browser hiển thị lại phía client. Khi người dùng yêu cầu 1 trang Asp, IIS server sẽ chuyển trang ASP đó cho một bộ phận xử lý gọi là ASP engine. Engine sẽ đọc mã nguồn file asp theo từng dòng, thực thi các script trong file. Cuối cùng file ASP được trả về cho người dùng dưới dạng một trang html thuần túy (không còn mã script) giống như trang web tĩnh. Nếu chúng ta xem lại mã nguồn của trang này trên browser thì có thể thấy những đoạn code asp trong file đã được dịch thành các dữ liệu html bình thường.



Hình 1.5 ASP engine xử lý file asp trước khi trả về cho browser



Hình 1.6 Trang ASP sau khi thực thi trả về cho client dưới dạng 1 trang web tĩnh.Browser không xem được mã nguồn của trang ASP

Bây giờ chúng ta quay lại bài toán Login ở trên. Ta có thể soạn thảo một trang Login.html và một trang Result.asp như sau:

Login.html

```
<html>
<head>
<title>New Page 1</title>
</head>
<body>
<form method="POST" action="Result.asp">
<p>Username: <input type="text" name="username" ></p>
<p><input type="submit" value="Submit" name="submit"></p>
</form>
</body>
</html>
```

Result.asp

```
<html>
<head>
<title>New Page 1</title>
</head>
<body>
<%
dim x
x=request.form("username") 'biến x nhận lại giá trị username từ form login
response.write "Hello "&x   'hiển thị nội dung tùy theo giá trị nhận được do
'người dùng điền vào form
%>
</body>
</html>
```

Một số ví dụ khác:
Hiển thị ngày giờ của server

```
<html>
<head>
```

<title>New Page 2</title>
</head>
<body>
<%response.write Now%>
</body>
</html>
Kết quả: 7/5/2005 12:21:57 PM
Hiển thị năm và tháng:
<%
response.write "Year: "&year(now)
response.write "Month:"&month(now)
%>
Kết quả: Year: 2005 Month:7

1.3 Tóm tắt các cú pháp VBScript

Mã lệnh ASP có thể viết bằng VBScript hoặc JavaScript (đọc thêm tài liệu về ngôn ngữ này). Các script của ASP thực thi trên server và nằm trong cặp dấu <% %>. Bên trong có thể chứa các biểu thức, hàm, toán tử, lệnh hợp lệ của ngôn ngữ Script tương ứng. Ở đây chúng ta tìm hiểu vắn tắt cách sử dụng ASP để lập trình web động bằng VBScript.

1.3.1 Response.write

để gửi nội dung về cho trình duyệt ta dùng lệnh Response.write
<%response.write "Hello World!"%>
hoặc có thể viết ngắn gọn hơn <%="Hello World!"%>

1.3.2 Biến

Biến dùng để lưu trữ thông tin. Biến có phạm vi cục bộ, nếu nó được khai báo bên trong 1 hàm hay thủ tục thì nó chỉ có tác dụng trong hàm hay thủ tục đó, nếu nó khai báo trong phạm vi toàn trang ASP thì tác dụng của nó sẽ có phạm vi trong toàn trang ASP, tuy nhiên không có tác dụng trong trang ASP khác.
Ví dụ ở trang Hello.asp ta có một biến x có giá trị là 3, trang Index.asp ta dùng lệnh <%response.write x %> thì sẽ không ra kết qủa là 3 vì biến x của trang Hello.asp không được hiểu trong trang Index.asp. Tương tự như vậy khi một biến được khai báo trong 1 hàm, sẽ không có tác dụng ở bên ngoài hàm đó.

Biến được khai báo và sử dụng bên trong trang asp nào dùng nó.
<%
Dim x 'khai báo biến, không bắt buộc
x=3
Response.write x
%>

Biến không bắt buộc phải khai báo.

Trong asp không khai báo kiểu của biến. Asp sẽ căn cứ vào việc sử dụng biến mà quyết định xem nên xử lý biến đó như là kiểu gì.

```
<%Dim a, b
a="Hello"          'a là  một biến kiểu chuỗi
For b=1 to 10      'b là một biến kiểu số nguyên
Response.write b
Next%>
```

Để có thể kiểm soát chính xác một biến theo kiểu mình mong muốn, chúng ta dùng các hàm chuyển đổi kiểu.

Để định nghĩa một biến có phạm vi sử dụng trong nhiều trang ASP của ứng dụng Web, ta dùng biến session và application (xem đối tượng session và application)

1.3.3 Mảng

Mảng dùng để lưu trữ dữ liệu theo một dãy các phần tử.

```
<%
dim y(5)      'khai báo mảng 6 phần tử đánh chỉ số từ 0 đến 5
y(0)=2
y(1)=13
response.write y(0)
response.write y(1)
%>
```

1.3.4 Ghép chuỗi

Để ghép các chuỗi với nhau ta dùng dấu &

```
<%Dim a, b
A="Cộng hòa xã hội chủ nghĩa Việt Nam"
B="Độc lập Tự do Hạnh phúc"
Response.write a&b
%>
```

1.3.5 Hàm có sẵn

VBScript hỗ trợ sẵn một số hàm cơ bản. Ví dụ hàm "now"sau đây sẽ trả về thời gian trên server

```
<%response.write now%>
```

1.3.5.1 Các hàm chuyển đổi kiểu

Các hàm này cho phép chuyển đổi kiểu dữ liệu

Cdate: Chuyển sang kiểu ngày tháng

```
<%Dim a, b
a="22/1/2004"          'a đang được hiểu là một chuỗi
b=Cdate(a)    'chuyển chuỗi a sang đúng kiểu ngày tháng
%>
```

Cint: Chuyển sang kiểu Integer

```
<% Dim a,b
a="3"
```

b=cint(a)
%>
Cstr: Chuyển sang kiểu string
<% Dim a,b
a=3
b=Cstr(a) %>
Các hàm khác : Cbyte, Cdbl,CSng, Cbool, Ccur,

1.3.5.2 Các hàm format
Các hàm này cho phép định dạng dữ liệu
FormatDateTime
FormatCurrency
FormatNumber
FormatPercent

1.3.5.3 Các hàm toán học:
Int: lấy phần nguyên của một số
<% Dim x=14.9
Y=Int(x) 'kết quả y=14
%>
Các hàm khác : Abs, Atn, Cos, Exp, Fix, Hex, Log, Oct, Rnd, Randomize, Round, Sin, Sqr, Tan

1.3.5.4 Các hàm thao tác với chuỗi
Len: Lấy chiều dài chuỗi
<%dim a,b
a="Cộng hòa xã hội chủ nghĩa Việt Nam"
b=len(a)
%>
Ucase, Lcase: Chuyển chữ hoa thành chữ thường và ngược lại
<%dim a,b,c,d
a="hello"
b=Ucase(a) 'b="HELLO"
c="GOODBYE"
d=Lcase(c) 'd="goodbye" %>
Ltrim, Rtrim, Trim: cắt bỏ các khoảng trắng thừa
<% dim a,b,c,d,e,f
a=" Hello"
b=Ltrim(a) 'cắt bỏ hết các khoảng trắng bên trái
c="Hello "
d=Rtrim(a) 'cắt bỏ hết các khoảng trắng bên phải
e=" Hello world "
f=Trim(a) 'cắt bỏ hết các khoảng trắng thừa 2 bên và ở giữa
%>
Left, Mid, Right: Lấy một chuỗi con trong chuỗi lớn

<%Dim a,b,c,d
a="Hello World"

b=left(a,5) 'lấy 5 ký tự bên trái của a, kết quả b="Hello"
c=right(a,5) 'lấy 5 ký tự bên phải của a, kết quả c="World"
d=mid(a,7,1) 'lấy 1 ký tự của a từ vị trí thứ 7, kết quả d="W"
%>

Các hàm khác: Space,String, StrReverse,StrComp,InStr,Replace,Split,join

1.3.5.5 Các hàm ngày tháng

Date, Time, Now: Lấy ngày, giờ hiện hành trên server

```
<%
Response.write "Hom nay la ngay: " &Date 'Date trả về ngày hiện hành
Response.write "Bay gio la"&Time  'Time trả về giờ  hiện hành
Response.write Now      'Now trả về ngày và giờ hiện hành
%>
```

Các hàm khác: DateAdd, DateDiff, DatePart, Year, Month, Day, Weekday, Hour, Minute, Second

1.3.5.6 Các hàm kiểm tra:

Các hàm này cho phép kiểm tra kiểu của biến và biểu thức

Isdate: Kiểm tra có phải đúng kiểu ngày tháng không?

```
<%Dim a
a="1/1/2004"
If Isdate(a) then
Response.write "a đúng là kiểu ngày tháng "
End if
%>
```

IsNumeric: Kiểm tra có phải đúng kiểu số không?

```
<%Dim a
A="13"
If IsNumeric(a) then
Response.write "a đúng là kiểu số"
End if
%>
```

Các hàm khác: IsArray,IsEmpty,IsNull,IsObject

1.3.6 Rẽ nhánh

1.3.6.1 If

Chúng ta sử dụng if theo cú pháp như ví dụ sau:

```
<% h=hour(now)
If h >12 then
Response.write "Afternoon"
else
Response.write "Morning"
End if %>
```

Hoặc:

```
<% h=hour(now)
```

If h >12 then Response.write "Afternoon" else Response.write "Morning" %>

1.3.6.2 Select case ... else ...End select

Cấu trúc rẽ nhánh trong trường hợp có nhiều hơn 2 lựa chọn

```
<%
h=hour(now)
Select case h
Case "1"
Response.write "1 am"
Case "2"
Response.write "2 am"
Case else
Response.write "Other "
End select
%>
```

1.3.7 Lặp:

1.3.7.1 For...Next

Vòng lặp có số lần lặp xác định

```
<%Dim i
For i=1 to 10
Response.write i
Next
%>
```

1.3.7.2 Do While...Loop

Vòng lặp có số lần lặp không xác định

```
<% Dim i
i=1
Do while i<=10
Response.write i
i=i+1
Loop
%>
```

1.3.7.3 While .. Wend

Vòng lặp có số lần lặp không xác định

```
<% Dim i
i=1
While i<=10
Response.write i
i=i+1
Wend
%>
```

1.3.7.4 Do .. Loop Until

Vòng lặp có số lần lặp không xác định

```
<%
```

```
i=1
do
response.write i
i=i+1
loop Until i>10
%>
```

1.3.8 Điều kiện and ,or, not

```
<% h=hour(now)
If (h >12) and (h<18) then
Response.write "Afternoon"
End if
%>
```

1.3.9 Thủ tục và hàm người dùng

Cũng như các ngôn ngữ lập trình khác, VBScript cho phép người dùng định nghĩa và sử dụng các thủ tục ,hàm. Nhờ vậy chương trình có thể chia thành các module nhỏ tạo nên cấu trúc lập trình sáng sủa (phương pháp chia để trị) Chẳng hạn với một bài toán ASP cần thực hiện việc hiển thị dữ liệu từ Database ra màn hình, ta có thể xây dựng các thủ tục hay hàm thực hiện từng nhiệm vụ đó:

- Thủ tục KetNoi
- Thủ tục HienThi
- Thủ tục HuyKetNoi

Như vậy phần chương trình chính sẽ rất sáng sủa, chúng ta chỉ việc gọi 3 thủ tục:

```
<%
KetNoi
HienThi
HuyKetNoi
%>
```

1.3.9.1 Thủ tục

Thủ tục thực hiện một nhóm các câu lệnh. Để viết một thủ tục chúng ta theo cấu trúc sau:

```
<%Sub TenThuTuc(Tham so)
' Phần thân của thủ tục
End Sub
%>
```

Ví dụ sau đây xây dựng chương trình đăng nhập gồm 2 file: Form.asp (hiển thị form để người dùng nhập username và password), Xulyform.asp (xử lý form, nếu username="test" và password="test" thì thông báo đăng nhập thành công, nếu không thì thông báo đăng nhập thất bại). File Xulyform.asp sẽ viết thủ tục và gọi thủ tục này:

Form.asp

```
<html>
<body>
<form method="post" action="xulyform.asp">
```

```
<input type="text" name="user">
<input type="password" name="pass">
<input type="submit" name="submit">
</form>
</body>
</html>
```

Xulyform.asp

```
<%Sub CheckUser(username,password)
if (username<>"test") or (password <> "test") then
response.write "Dang nhap that bai!"
 else
response.write "Dang nhap thanh cong!"
end if
End Sub
%>
<% dim a, b
a=request.form("user")
b=request.form("pass")
CheckUser a,b                    'gọi thủ tục
%>
```

1.3.9.2 Hàm

Hàm khác với thủ tục là nó trả về một kết quả. Để viết một hàm chúng ta viết theo cấu trúc sau:

```
<%Function TenFunction(tham so)
' Phần nội dung của hàm
End Function
%>
```

Chú ý trong nội dung của hàm bao giờ cũng phải có một lệnh trả về kết quả: TenFunction=...

Với bài toán đăng nhập ở trên chúng ta có thể viết lại như sau (file xulyform.asp dùng hàm)

Form.asp

```
<html>
<body>
<form method="post" action="xulyform.asp">
<input type="text" name="user">
<input type="password" name="pass">
<input type="submit" name="submit">
</form>
</body>
</html>
```

Xulyform.asp

```
<%Function CheckUser(username,password)
if (username<>"test") or (password <> "test") then
CheckUser="False"
 else
```

```
 CheckUser="True"
end if
End Function
%>
<%
dim a
a=CheckUser(request.form("user"),request.form("pass"))    ' gọi hàm
if a="True" then
response.write "Dang nhap thanh cong"
else
response.write "Dang nhap that bai"
end if
%>
```

1.3.10 Sử dụng #include

Trong trường hợp muốn trộn mã nguồn từ 1 file asp vào 1 file asp khác trước khi server thực thi nó, người ta dùng thẻ định hướng #include với cú pháp
<!--#include file="Tenfile"-->
Một số ứng dụng của #include như người ta thường include file chứa các hàm thư viện dùng chung cho cả ứng dụng vào đầu file Asp nào cần sử dụng thư viện này, hoặc insert các file Header và Footer cho 1 trang web, insert các thành phần được sử dụng chung trong nhiều file asp như menu,...
Ví dụ trong ứng dụng ASP có nhiều trang cần thao tác với database, chúng ta sẽ viết riêng module thao tác với database ra một file myConnection.asp, rồi include file này vào trang asp nào muốn thao tác với database

```
<!--#include file="myConnection.asp"-->
<%
' mã nguồn
%>
```

Lưu ý là include file được thực hiện trước khi script chạy. Vì vậy đoạn lệnh sau đây là không hợp lệ:

```
<% filename="myConnection.asp"%>

<!--#include file="<%=filename%>"-->
```

1.4 Các đối tượng căn bản

Đối tượng là một nhóm các hàm và biến. Một số đối tượng đã được xây dựng sẵn và có thể sử dụng ngay mà không cần khởi tạo: Request, Response, Session, Application, Server. Một số đối tượng cần khởi tạo nếu muốn sử dụng Dictionary, Connection, Recordset...

1.4.1 Đối tượng Request

Request và Response là 2 đối tượng được dùng nhiều nhất trong lập trình ASP, dùng trao đổi dữ liệu giữa trình duyệt và server.
Request cho phép lấy về các thông tin từ client. Khi browser gửi một yêu cầu trang web lên server ta gọi là 1 request

Chúng ta thường sử dụng các lệnh request sau:

1.4.1.1 Request.QueryString

Cho phép server lấy về các giá trị được gửi từ người dùng qua URL hoặc form (method GET).

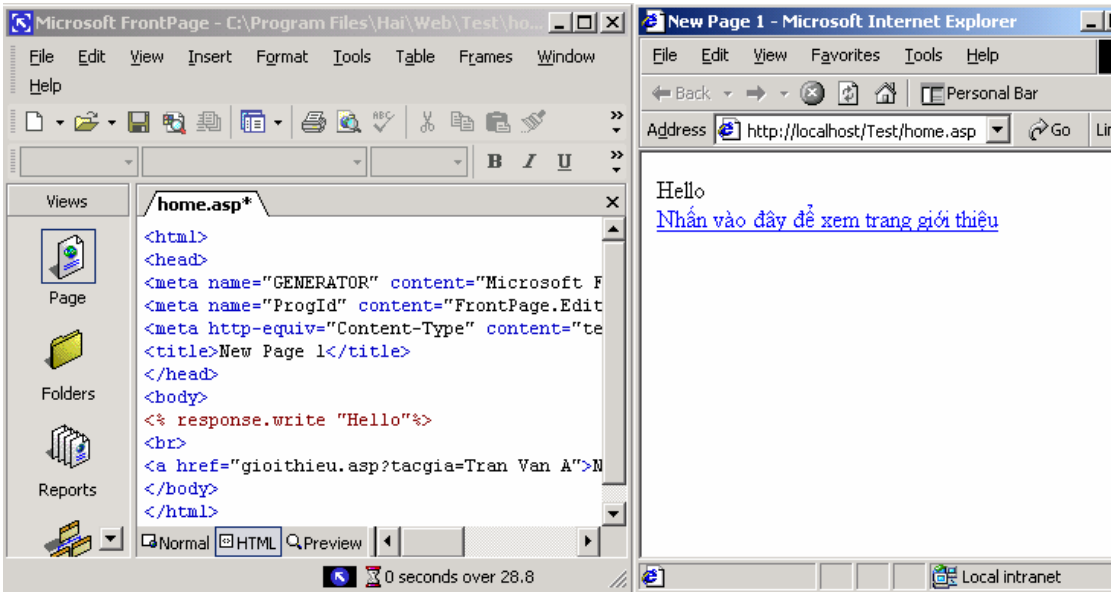
Ví dụ ở trang home.asp chúng ta đặt một dòng liên kết sang trang gioithieu.asp với thẻ sau:

Nhấn vào đây để sang trang giới thiệu

biến "tacgia" có giá trị là "Tran Van A" được người dùng gửi tới server kèm theo URL. (người dùng có thể gõ thẳng địa chỉ "http://localhost/alias/gioithieu.asp?tacgia=Tran Van A" trên thanh Address của trình duyệt)

Server muốn nhận lại giá trị này thì dùng **request.QueryString** ở trang gioithieu.asp

```
<%dim a
a=request.querystring("tacgia")    'lúc này a có gía trị là "Tran Van A"
response.write "Tác giả của trang home.asp là: " &a
%>
```



Hình 1.7

Tương tự như vậy nếu người dùng gửi giá trị Tran Van A thông qua một biến trong form và chọn method GET

```
<form method="get" action ="gioithieu.asp">
<input type="text" name="tacgia" value="Tran Van A">
<input type="submit" name="submit" value="Nhan vao day de sang trang gioi thieu">
</form>
```

1.4.1.2 Request.Form
Cho phép server lấy về các giá trị được gửi từ người dùng qua form (method POST).

Chẳng hạn file **form.asp**:
<form method="POST" action ="xulyform.asp">
<input type="text" name="User">
<input type="submit" name="submit" value="Nhan vao day de sang trang gioi thieu">
</form>

File xulyform.asp làm nhiệm vụ xử lý thông tin từ Form này sẽ dùng câu lệnh request.form để nhận lại thông tin người dùng đã gõ vào:
<%Dim x
x=Request.form("User") %>
response.write "Tên người dùng là: "&x
%>

1.4.2 Response
Đối tượng Response dùng để gửi các đáp ứng của server cho client.
Chúng ta thường dùng một số lệnh Response sau:
1.4.2.1 Response.Write
Đưa thông tin ra màn hình trang web
Ví dụ để đưa câu chào Hello ra màn hình ta dùng lệnh sau:
<%response.write "Hello"%>
Hiển thị thời gian trên server ra màn hình:
<%response.write now%>
hoặc *<%=now%>*
now là hàm lấy ngày giờ hệ thống trên server
1.4.2.2 Response.Redirect
Chuyển xử lý sang một trang Asp khác.
Ví dụ trang xulyform.asp sau khi kiểm tra form đăng nhập thấy người dùng không có quyền vào website thì nó sẽ chuyển cho file Error.asp(file này hiển thị một thông báo lỗi user không có quyền truy cập)
<% Response.redirect "error.asp" %>
1.4.2.3 Response.End
Ngừng xử lý các Script. Dùng lệnh này khi muốn dừng xử lý ở một vị trí nào đó và bỏ qua các mã lệnh ASP ở phía sau. Đây là cách rất hay dùng trong một số tình huống, chẳng hạn như debug lỗi

1.4.3 Đối tượng Session
Session là một phiên làm việc giữa từng người dùng và web server, nó bắt đầu khi người đó lần đầu tiên truy cập tới 1 trang web trong website và kết thúc khi người đó rời khỏi website hoặc không tương tác với website trong một khoảng thời gian nhất định (time out). Như vậy tại một thời điểm một website có bao nhiêu người truy cập thì có bấy nhiêu phiên ứng với mỗi người, các phiên này độc lập nhau. Để lưu những thông tin tác dụng trong 1

phiên, người ta dùng đối tượng Session, ví dụ khi một user bắt đầu session với việc login vào hệ thống, và user đã login đó cần được hệ thống ghi nhớ trong toàn phiên làm việc (nhằm tránh việc người dùng phải đăng nhập lại mỗi khi đưa ra một request).Giá trị của biến kiểu session có phạm vi trong tất cả các trang ASP của ứng dụng, nhưng không có tác dụng đối với phiên làm việc khác.
Ví dụ, sử dụng biến session sau đây đếm số lần 1 người đã truy cập vào trang web:

Home.asp

<% session("x")=session("x")+1 %>

session("x") đại diện cho số lần mà một user đã truy cập vào trang home.asp. Với 2 người dùng khác nhau thì giá trị session("x") lại khác nhau. Thật vậy , A có thể truy cập 10 lần (session("x") =10) trong khi B có thể truy cập 2 lần thôi (session("x") =2)

Server kết thúc và hủy bỏ đối tượng session khi:
- Người dùng không triệu gọi các trang của ứng dụng hoặc cập nhật làm mới (refresh) lại thông tin của trang trong một thời gian nhất định. Khi một session hết thời gian hiệu lực nó sẽ được xem như hết hạn sử dụng ,tất cả các biến lưu trong session và bản thân session sẽ bị hủy bỏ. Có thể kiểm tra và tăng giảm thời gian Timeout của Session tính bằng giây như sau:

```
<%
      Session.Timeout = 500
%>
```

- Trang ASP gọi đến phương thức Abandon của Session .

```
<%
 Session.Abandon
 %>
```

Việc khởi tạo và kết thúc 1 biến session có thể viết trong các hàm sự kiện Session_OnStart và Session_OnEnd được định nghĩa trong file global.asa

1.4.4 Đối tượng Application

Application đại diện cho toàn bộ ứng dụng, bao gồm tất cả các trang web trong website. Để lưu trữ những thông tin có tác dụng trong toàn ứng dụng, tức là có giá trị trong tất cả các trang asp và tất cả các phiên, người ta dùng đối tượng Application
Điểm khác của biến application so với biến session là session chỉ có tác dụng đối với mỗi phiên, còn biến application có tác dụng với mọi phiên.
Ví dụ, để đếm xem có bao nhiêu người truy cập vào website, chúng ta có thể dùng một biến Application. Mỗi khi một người dùng mới truy cập vào website ta tăng biến này lên 1 đơn vị để chỉ rằng đã có thêm 1 người truy cập.

<% application("x")=application("x")+1 %>

Trang home.asp muốn hiển thị số người truy cập chỉ cần in giá trị của biến này

```
<% response.write "Số người đã truy cập vào website là:"&application("x")
 %>
```

Với 2 phiên khác nhau thì giá trị application("x") là như nhau. Thật vậy , A và

B khi truy cập vào trang home.asp đều thấy: "Số người đã truy cập vào website là 3" (trong trường hợp application("x") =3)
Việc khởi tạo và kết thúc 1 biến application có thể viết trong các hàm sự kiện Application_onStart và Application_onEnd được định nghĩa trong file global.asa

Khóa Application:
Do biến application có thể được dùng chung bởi nhiều phiên nên sẽ có trường hợp xảy ra xung đột khi có 2 phiên cùng thay đổi giá trị một biến application. Để ngăn chặn điều này chúng ta có thể dùng phương thức Application.lock để khóa biến application trước khi thay đổi nó. Sau khi sử dụng xong biến này có thể giải phóng khóa bằng phương thức application.unlock (xem ví dụ sau).

1.4.5 File Global.asa

File này là file tùy chọn chứa các khai báo đối tượng, biến có phạm vi toàn ứng dụng. Mã lệnh viết dưới dạng Script. Mỗi ứng dụng chỉ được phép có nhiều nhất 1 file Global.asa, nằm ở thư mục gốc của ứng dụng. Người ta thường dùng global.asa trong trường hợp muốn có những xử lý khi một session bắt đầu hay kết thúc, một application bắt đầu hay kết thúc, thông qua các hàm sự kiện :
Application_Onstart : hàm sự kiện này xảy ra khi ứng dụng asp bắt đầu hoạt động, tức là khi người dùng đầu tiên truy cập tới trang web đầu tiên khi ứng dụng hoạt động.

Session_Onstart: hàm sự kiện này xảy ra mỗi khi có một người dùng mới truy cập vào ứng dụng (bắt đầu 1 session)

Session_OnEnd: hàm sự kiện này xảy ra mỗi khi 1 người dùng kết thúc session của họ
Application_OnEnd: hàm sự kiện này xảy ra khi ứng dụng dừng.

File **Global.asa** có cấu trúc như sau:

```
<script language="vbscript" runat="server">
Sub Application_OnStart
'.........
End sub
Sub Application_OnEnd
'.............
End Sub
Sub Session_OnStart
'.........
Application("x")=Application("x")+1
End sub
Sub Session_OnEnd
'.............
End Sub
</script>
```

Ví dụ sau đây sẽ đếm số người dùng hiện đang truy cập website. Số người dùng được lưu trữ trong biến Application(“songuoi”). Ở bất cứ đâu trong ứng dụng nếu muốn hiển thị số người dùng chúng ta chỉ việc chèn lệnh hiển thị nó:

<%=Application(“songuoi”)%>

Ngoài ra ứng dụng cũng cho phép đếm số lần 1 người đã truy cập website trong phiên làm việc của họ. Số lần được lưu trữ trong biến Session(“solan”)

Global.asa

```
<script language="vbscript" runat="server">
Sub Application_OnStart
Application("songuoi")=0
End Sub

Sub Session_OnStart
Application.Lock
Application("songuoi")=Application("songuoi")+1
Application.UnLock

Session(“solan”)=0
End Sub

Sub Session_OnEnd
Application.Lock
Application("songuoi")=Application("songuoi")-1
Application.UnLock
End Sub

Sub Application_OnEnd
End Sub

</script>
```

Home.asp

```
<html>
<body>
<p>
Có <%response.write(Application("songuoi"))%>
người đang truy cập website
</p>
<%session(“solan”)= session(“solan”)+1 %>
<p>
Bạn đã truy cập trang này <%response.write(session("solan"))%>
lần!
</p>
</body>
</html>
```

1.4.6 Đối tượng Dictionary

Đối tượng Dictionary lưu trữ thông tin theo từng cặp khóa/ giá trị. Nó khá giống với mảng nhưng có khả năng xử lý linh hoạt đối với những cặp dữ liệu có quan hệ kiểu từ điển (cặp khóa/ giá trị ví dụ như : mã Sinh viên/ tên Sinh viên), trong đó khóa được xem là từ cần tra và giá trị chính là nội dung của từ tra được trong từ điển.

Muốn sử dụng đối tượng Dictionary chúng ta phải khởi tạo nó:

```
<%set d=server.createObject("Scripting.Dictionary")
d.add "work","Lam viec"
d.add "learn","Hoc tap" 'tương tự như mảng nhưng mỗi phần tử là một cặp khóa/giá trị
response.write "work nghĩa  tiếng Việt là: "&d.item("work")
response.write "learn nghĩa  tiếng Việt là: "&d.item("learn")
set d=nothing
%>
```

Một số ứng dụng của đối tượng này như dùng mô phỏng giỏ hàng chứa hàng hóa(shopping cart) với cặp khóa/giá trị là :ProductID/Quantity (xem chương 2), sổ địa chỉ với cặp khóa/giá trị là: CustomerName/Address.

1.4.7 Đối tượng Server

Đối tượng Server được dùng để truy cập các thuộc tính và phương thức của server .Ta thường dùng 2 lệnh sau

1.4.7.1 Server.CreateObject

khởi tạo 1 đối tượng.

Ví dụ:

Tạo một đối tượng Connection:

```
<%Set conn=Server.CreateObject("ADODB.Connection")%>
```

Tạo một đối tượng Dictionary:

```
<%set d=server.createObject("Scripting.Dictionary")%>
```

1.4.7.2 Server.Mappath

biến đường dẫn tương đối thành tuyệt đối.

Ví dụ:

```
<%str= server.mappath("nhanvien.mdb")
Response.write str%>
```

Sẽ cho kết quả: "C:\WEB\nhanvien.mdb" trong trường hợp file nhanvien.mdb nằm trong thư mục C:\WEB

Ta thường áp dụng server.mappath trong những trường hợp xử lý đường dẫn tương đối, ví dụ là chuỗi kết nối vào database

```
connstr="provider=microsoft.jet.oledb.4.0; data
source="&server.mappath("nhanvien.mdb")&";"
```

1.5 Sử dụng Database với ASP

Hầu hết các ứng dụng Web động đều lưu trữ dữ liệu trong Database. Vì vậy các thao tác kết nối vào Database, xem, thêm, sửa, xóa dữ liệu trong các bảng là phần quan trọng đối với các ngôn ngữ lập trình web như ASP.

Chúng ta sẽ học các kỹ thuật sử dụng Asp để thao tác với dữ liệu trong Database thông qua kiến trúc ADO.

1.5.1 Các cú pháp căn bản để truy xuất dữ liệu từ DB

Để thao tác với dữ liệu trong các bảng của DB, có 4 thao tác chính với câu lệnh SQL tương ứng như sau:

(Lấy ví dụ với một Database cụ thể Quanlyhocvien.mdb, trong đó có một bảng HosoHocVien (MaHV:text, Ten: text)

1.5.1.1 Lựa chọn

Lấy tất cả các bản ghi trong bảng:
*"Select * from HosoHocVien"*
Nếu lựa chọn có điều kiện:
*"Select * from HosoHocVien where MaHV='10' "*
Nếu chỉ lựa chọn một số trường trong bảng:
"Select Ten from HosoHocVien where MaHV='10' "

1.5.1.2 Thêm dữ liệu vào bảng

"Insert into HosoHocVien values ('001','Tran Van A') "

1.5.1.3 Sửa dữ liệu

"Update HosoHocVien set Ten='Tran Van B' where MaHV='001' "

1.5.1.3 Xoá dữ liệu

"Delete from HosoHocVien where MaHV='001' "

Chúng ta có thể sử dụng các lệnh SQL phức tạp hơn để có được kết quả mong muốn như sử dụng các lệnh join, order by, group by, having...

1.5.2 Đối tượng Connection

Đối tượng Connection cho phép tạo kết nối đến một DB.
Các bước sử dụng Connection:

- Khai báo đối tượng Connection
- Khởi tạo
- Tạo chuỗi kết nối
- Mở Connection với chuỗi kết nối trên
- Sử dụng Connection
- Đóng và Hủy Connection

Ví dụ sau đây kết nối đến database Access QuanlyHocvien.mdb (database này nằm trong cùng thư mục với file Asp)

```
<%
dim conn          'khai báo
set conn=server.createObject("ADODB.connection")  'khởi tạo
stringconn="provider=microsoft.jet.OLEDB.4.0;data
source="&server.mappath("QuanlyHocVien.mdb")&";"      'chuỗi kết nối
conn.open stringconn     'mở connection
' các thao tác với DB sử dụng connection này
'.......
'conn.close              'đóng connection
Set conn=nothing      'hủy connection
%>
```

(chuỗi "stringconn=…" viết trên 1 dòng, trong đó: "… data source = …"chú ý có một dấu cách giữa "data" và "source", chuỗi này chỉ đúng với Access)

1.5.3 Đối tượng Recordset

Đối tượng Recordset thường dùng để xem, thêm, sửa, xóa các bản ghi trong bảng dữ liệu của Database. Nó trỏ đến tập hợp các bản ghi là kết quả trả về từ câu lệnh select
Các bước sử dụng đối tượng Recordset :

- Khai báo đối tượng Recorset
- Khởi tạo
- Tạo sql query
- Mở Recordset với chuỗi sql query và connection đã mở
- Sử dụng Recordset
- Đóng và Hủy Recordset

Ví dụ sau đây cho phép lấy các bản ghi trong bảng và hiển thị ra ngoài trang web.

```
 <%Dim rs ' khai báo Recordset
set rs=server.createObject("ADODB.Recordset")        'Khởi tạo
SQLstring="select * from HosoHocVien"                'SQL query
rs.open SQLstring ,conn                    'Mở Recordset
' dùng vòng lặp để hiển thị toàn bộ các bản ghi ra màn hình
do while not rs.EOF
        response.write RS("MaHV")
        response.write RS("Ten")
        response.write "<BR>"
        rs.movenext  'dịch con trỏ rs tới bản ghi tiếp theo
loop
rs.close 'đóng recordset
set rs=nothing 'hủy recordset
%>
```

Chúng ta có thể kết hợp giữa script và thẻ html để dữ liệu được hiển thị ra ngoài trang web với giao diện theo ý muốn :

```
  <table border="1">
    <tr>
      <td>MA HOC VIEN</td>
      <td>TEN</td>
    </tr>
<%do while not rs.eof%>
    <tr>
      <td ><%=rs("MaHV")%></td>
      <td ><%=rs("Ten")%></td>
```

```
    </tr>
<%rs.movenext
loop
rs.close
%>
  </table>
```

Sau đây là một ví dụ hoàn chỉnh liệt kê các user trong bảng tblUser ra trang web:

Connection.asp

```
<%
dim conn
Sub openConn()
set conn=server.createobject("adodb.connection")
connstr="provider=microsoft.jet.oledb.4.0;                    data
source="&server.mappath("myDB.mdb")&";"
conn.open connstr
End Sub

Sub destroyConn()
conn.close
set conn=nothing
End Sub
%>
```

ListUser.asp

```
<!--#include file ="Connection.asp"-->
<%openConn
set rs = server.createobject("ADODB.Recordset")
rs.open "select * from tblUser", conn%>
<table border="1" width="200">
        <tr><td>ID</td><td>Username</td><td>Address</td>
<% do        while not rs.EOF
<tr>
<td><%=rs("id")%></td>
<td><%=rs("username")%></td>
<td><%=rs("address")%></td>
</tr>
<%    rs.movenext
loop
rs.close
destroyConn%>
</table>
```

1.5.4Thêm sửa xóa dữ liệu trong DB:

Với một connection đã mở chúng ta có thể dùng nó để thực thi câu lệnh SQL dạng insert, update, delete:

Thêm dữ liệu:

```
<%Conn.execute "Insert into HosoHocvien values('001','Tran Van A')"%>
```

Sửa dữ liệu:

<%Conn.execute "Update HosoHocVien set Ten='Tran Van B' where MaHV='001' "%>

Xoá dữ liệu:

<%Conn.execute "Delete from HosoHocVien where MaHV='001' " %>

Ngoài ra chúng ta có thể dùng Recordset để thêm, sửa, xóa dữ liệu trong database bằng cách duyệt qua tập hợp các bản ghi trong bảng

Thêm dữ liệu:

```
<%Dim RS
set rs=server.createObject("ADODB.recordset")
SQLstring="select * from HosoHocVien"
rs.open SQLstring ,conn,3,2
'rs.open SQLstring ,conn,adOpenStatic,adLockPessimistic
rs.addnew                    'Thêm một bản ghi
rs("MaHV")="001"             ' gán giá trị cho các trường của bản ghi
rs("Ten")="Tran Van A"
rs.update                    ' Xác nhận thêm xong
rs.close                     'đóng recordset
%>
```

Sửa:

```
<% set rs=server.createObject("ADODB.recordset")     'Khởi tạo
SQLString="select * from HosoHocVien where ma='001' "
' lấy ra bản ghi cần sửa
rs.open SQLString ,conn,3,2
rs("Ten")="Tran Van B"           'sửa lại giá trị trường "Ten"
rs.update                        ' xác nhận sửa xong
rs.close                         'đóng recordset
%>
```

Xóa:

```
<% set rs=server.createObject("ADODB.recordset")     'Khởi tạo
SQLString="select * from HosoHocVien where MaHV='001' "     'Câu lệnh
SQL lấy ra đúng bản ghi cần xóa
rs.open SQLString ,conn,3,2
rs.delete                'xóa bản ghi này
rs.close             'đóng recordset
%>
```

1.5.4 Phân trang

Trong nhiều trường hợp do kết quả câu lệnh "select ...." trả về quá nhiều bản ghi, nếu chúng ta hiển thị tất cả trên cùng 1 trang web thì sẽ bất tiện trong việc đọc chúng, khi đó người ta tiến hành phân nó ra để hiển thị thành nhiều trang, đây gọi là kỹ thuật phân trang. So với cách đọc và hiển thị dữ liệu

thông thường, thì phân trang đòi hỏi phải thiết lập thêm một số thuộc tính:

- Số bản ghi cần hiển thị trên một trang RS.PageSize
- Trang nào đang được hiển thị: RS.AbsolutePage,
- Khi mở Recordset đòi hỏi phải thêm các tham số CursorType và LockType :rs.open SQLstring ,conn,3,3
- Vòng lặp hiển thị dữ liệu cần có cơ chế đảm bảo nó chỉ chạy đúng số bản ghi trên một trang (rs.pagesize) là phải thoát khỏi vòng lặp.

Ví dụ để hiển thị bảng HosoHocVien với yêu cầu chỉ hiển thị 4 bản ghi/1 trang:

Home.asp

```
<%
dim x          'biến này dùng để xác định xem cần hiển thị trang nào
x=request.querystring("PageNumber")      'nhận lại PageNumber khi người dùng nhấn vào các nút "Trước" và "Tiếp"
if x="" then            'đầu tiên sẽ hiển thị trang 1
x=1
end if

dim conn
set conn=server.createObject("ADODB.connection")
stringconn="provider=microsoft.jet.OLEDB.4.0;data
source="&server.mappath("QuanlyHocVien.mdb")&";"
conn.open stringconn
Dim RS
set rs=server.createObject("ADODB.recordset")
SQLstring="select * from HosoHocVien"
rs.pagesize= 4    'chỉ hiển thị 4 bản ghi/1 trang
rs.open SQLstring ,conn,3,3
rs.AbsolutePage=x                              'trang cần hiển thị
dem=0          'biến này để đảm bảo vòng lặp chỉ thực hiện tối đa 4 lần lặp
do while not rs.EOF  and dem<rs.pagesize
        response.write RS("MaHV")
        response.write RS("Ten")
        response.write "<BR>"
dem=dem+1
rs.movenext
loop
%>
<%   'Hiển thị nút "Trước"
if  x>1 then          %>
<a href="home.asp?pageNumber=<%=x-1%>">Trước</a>
<%end if%>
<%   'Hiển thị nút "Tiếp"
if  not RS.EOF then          %>
<a href="home.asp?pageNumber=<%=x+1%>">Tiếp</a>
```

```
<%end if
rs.close 'đóng recordset
%>
```

1.5.5 Tìm kiếm dữ liệu trong database

Để tìm kiếm dữ liệu trong bảng của Database chúng ta dựa vào câu lệnh SQL:

*"select * from Tenbang where Tencot like '%giatri%' "*

Ví dụ đoạn chương trình sau cho phép hiển thị những Sinh Viên trong bảng "HosoHV" của DB "Sinhvien.mdb" có tên được tìm kiếm bởi từ khoá "Anh" (Ví dụ : Tuấn Anh, Vân Anh, Việt Anh...)

```
<%
set conn=server.createobject("adodb.connection")
connstring="provider=microsoft.jet.oledb.4.0;data
source="&server.mappath("sinhvien.mdb")&";"
conn.open connstring
set rs=server.createobject("adodb.recordset")
rs.open "select * from HosoSV where ten like '%Anh%' ",conn
do while not rs.eof
response.write rs("MaSV")
response.write "  "
response.write rs("Ten")
response.write "  "
response.write rs("Lop")
response.write "<BR>"
rs.movenext
loop
rs.close
%>
```

Thông thường người sử dụng nhập từ khoá cần tìm kiếm vào một trường của form. như vậy ta chỉ việc dùng lệnh request.form để lấy lại từ khoá cần tìm kiếm và đưa vào câu lệnh SQL ở trên. Chẳng hạn người sử dụng nhập từ khoá cần tìm vào trường "Ten" trong form thì chúng ta sẽ mở bảng bằng câu lệnh SQL sau:

```
<% ten=request.form("Ten")
'validate
rs.open "select * from HosoSV where Ten like '%"&ten&"%' ",conn
'... %>
```

Nếu không tìm thấy bản ghi nào thì giá trị rs.EOF sẽ true.

```
<%
If rs.eof then response.write "Không tìm thấy kết quả nào" %>
```