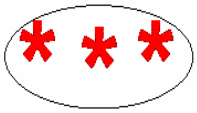
Mariano Pirrotta

Pagine ASP

Ac tive **S**erver **P**age



Volume 4

Web server

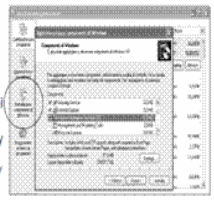
- software per la gestione di un computer **host** Internet oppure di un **server di rete**, che mette a disposizione dati o applicativi per gli utenti della rete

Web server

- **IIS** (*Internet Information Services*), Microsoft, per sistemi Windows (non disponibile per Windows Home e Millenium)
- **Apache**, Open Source e licenza GNU, per sistemi Linux, ma anche Windows (***a patchy server***)

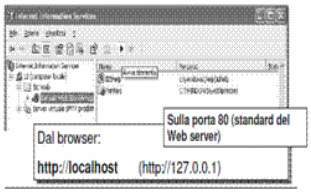
Installazione di IIS

- Start, Impostazioni, Pannello di controllo, Installazione applicazioni, Installazione componenti di Windows.
- Crea directory **Inetpub** e sottodirectory **wwwroot**



Avvio e arresto del Web server

- Pannello di controllo, Strumenti di amministrazione, IIS



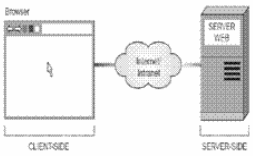
Per visualizzare le pagine Web

- No doppio click sul file, ma ...
- http://localhost/pagina.htm

oppure

- http://localhost/pagina.asp

Server e client



Pagine ASP

- HTML + linguaggio ASP (Visual Basic o C#)
- ASP classiche
- ASP.NET (.NET Framework)
- Per database: librerie e oggetti ADO e ADO.NET

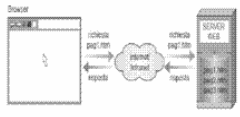
Strumenti per sviluppo software

- Blocco note
- Web editor (Dreamweaver)
- Visual Web Developer 2005 Express (funziona anche senza IIS, rende disponibile un **server di sviluppo** su una porta diversa da 80; per esempio: http://localhost:1094)

DEMO

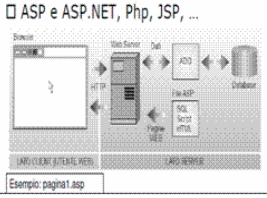
Tecnologie lato client

- Esecuzione sul computer dell'utente
- HTML, JavaScript, Applet, Flash, ...



Tecnologie lato server

- Esecuzione sul server e creazione dinamica della pagina Web
- ASP e ASP.NET, Php, JSP, ...



Risorse Web

- www.w3schools.com
- www.html.it
- http://www.asp.net/
- .....
- .....

Richiami sul linguaggio HTML

```
<html>
<head>
<title>            </title>
</head>
<body>


</body>
</html>
```

Form HTML

```
<FORM NAME="form1" ACTION="pagina1.asp"
METHOD="post">

Nome: <INPUT TYPE="text" NAME="nome" /><BR />

E-mail: <INPUT TYPE="text" NAME="email"
/><BR>

<INPUT TYPE="submit" VALUE="Invia" NAME="b1">

<INPUT TYPE="reset" VALUE="Annulla"
NAME="b2">

</FORM>
```

Il linguaggio ASP (1)

- Delimitatori di codice ASP
  <% %>
- <%@ LANGUAGE = VBScript %>
- <% Option Explicit %>
- ' Riga1 di commento
- Dim NomeVariabile
- A = 3

Il linguaggio ASP (2)

- Dim A(5)
- A(0) = 345
- If (condizione) Then
  istruzioni1
  Else
  istruzioni2
  End If

Il linguaggio ASP (3)

- Select Case Selettore
  Case valore1, valore2, valore3
  istruzioni1
  Case valore4, valore5
  istruzioni2
  Case Else
  istruzioni3
  End Select

Il linguaggio ASP (4)

- Dim i
  For i = ValoreIniziale to ValoreFinale
  istruzioni
  Next
- Do While(condizione)
  istruzioni
  Loop

Oggetti ASP (1)

- Oggetto.Metodo *parametri*
  Oggetto.Proprietà
- Oggetto **Request:** per utilizzare le informazioni provenienti dal browser dell'utente (form)

```
Nome = Request.Form("studente")
```

Oggetti ASP (2)

- Oggetto **Response** per inviare dati al browser dell'utente
  `Response.Write ("Hello " & Nome)`
- Forma abbreviata:
  `<% ="Hello " & Nome %>`

Esempio: pagina2.htm + pagina2.asp

Oggetti server

- per creare un oggetto tra quelli contenuti nelle librerie software (dette **componenti**):
  Server.CreateObject
- Esempio:

```
conn = Server.CreateObject("ADODB.Connection")
```

5

6

COMANDI ASP

Le pagine ASP (*Active Server Pages*) sono una tecnologia, sviluppata da Microsoft, mediante la quale è possibile realizzare pagine Web dinamiche il cui codice viene processato dal Server Web prima di essere inviato al client.

Le pagine ASP sono file di testo: la loro creazione può essere fatta utilizzando Blocco Note o un qualunque pacchetto applicativo per la creazione di progetti Web. Esse possono contenere diversi elementi:

- dichiarazione del linguaggio di scripting (VBScript è utilizzato di deafult);
- comandi ASP;
- codice HTML;
- codice in linguaggio specifico quando si accede a un database.

I file che contengono comandi ASP vengono salvati con estensione **.asp** e sono pubblicate sul Server Web presso il quale i client faranno richieste di servizi; sul server le pagine devono essere salvate nella cartella riconosciuta come WWWROOT.

Computer > Disco locale(C:) > inetpub > wwwroot

COMANDI ASP

Comunicare con i form

I form	• **inviano dati al server**	Metodo get Metodo post
Gli oggetti del form sono	• **caselle di testo singole** • **pulsanti di azione** • **pulsanti di opzione singola** • **pulsanti di opzione multipla** • **i menu a tendina** • **l'area di testo**	
I comandi ASP	• **realizzano pagine Web dinamiche lato server** • **contengono**	Dichiarazione linguaggio script Codice HTML Codice script Codice linguaggio per data base
Gli oggetti ASP sono	• **Request** • **Response** • **Server** • **Application** • **Session** • **ObjectContext e Transaction Server**	
I metodi ASP sono	• **l'oggetto Request**	Querystring Form
	• **l'oggetto response**	Write Redirect End
Il server riceve i dati del form	• **con il metodo get**	Request.querystring
	• **con il metodo post**	Request.form

8

COMANDI ASP

Come provare il codice ASP sul proprio computer

È possibile provare il codice ASP anche sul proprio computer simulando la creazione di un'architettura client/server attraverso il sistema operativo Windows; dopo aver installato l'applicazione IIS come definito nell'unità precedente dobbiamo procedere nel modo seguente:

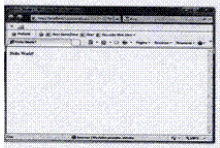
- aprire Blocco Note e scrivere il codice della pagina;
- salvare il file con estensione .asp nella Home Directory (di default sui sistemi Windows si chiama **C:\Inetpub\wwwroot**);
- aprire il browser normalmente utilizzato e digitare l'indirizzo **http://localhost/nomefile.asp**;
- comparirà una pagina eseguita dal server simulato.

Sintassi di una pagina ASP

Ogni pagina ASP inizia con l'indicazione del linguaggio di script utilizzato attraverso il tag:

```
<%@language=NomeDelLinguaggioScript%>
```

Se l'utente utilizza VBScript il comando non è necessario in quanto è lo script utilizzato di default da ASP, altrimenti esso deve sempre comparire nella prima riga.
Tutti i comandi ASP devono essere delimitati dai tag **<%.....%>**.
Per esempio una semplice pagina ASP può essere la seguente: il server invia al client una pagina Web nella quale si visualizza la stringa **"Hello World!"**; il file si chiama **ciaomondo.asp**, il nome è chiaramente visibile sulla barra dell'indirizzo ed è stata pubblicata sul server simulato **http://localhost**.



```
<%@Language=VBScript%>  DICHIARAZIONE LINGUAGGIO
<Html>
  <Head>
    <Title>Hello World !</Title>    CODICE HTML
  </Head>
  <Body>
    <%Response.Write "Hello World!"%> CODICE ASP
  </Body>
</Html>
```

Glossario

il **codice sorgente** è un applicativo scritto in linguaggio di programmazione ad alto livello.

Quando utilizziamo le pagine ASP è molto importante notare la differenza tra il codice sorgente e codice ricevuto dal browser:
Se clicchiamo sul menu **Visualizza → HTML** notiamo che la pagina che si apre è pagina molto simile a quella scritta in precedenza ma si può notare una differenza importantissima: all'interno di questa pagina non compaiono i caratteri <% e %> ciò significa che non è stato inviato al browser il sorgente della **lezione1.asp** ma solo codice HTML, reinterpretato.

Reinterpretazione significa che il server prima di inviare al client il file ASP riscrive la pagina sostituendo quello che c'è all'interno dei caratteri <%.....%> con la sua elaborazione.

Questo è il meccanismo fondamentale su cui si basa la tecnologia ASP.

COMANDI ASP

Gli oggetti ASP

Un **oggetto** rappresenta un elemento del linguaggio di programmazione caratterizzato da proprietà e metodi.
Le **proprietà** sono le caratteristiche dell'oggetto, i **metodi** sono le azioni che è in grado di eseguire.
Nella tecnologia ASP esistono sei oggetti (*Active Server Objects*), ognuno dei quali gestisce un particolare aspetto dell'interazione client/server. Essi sono:

Request	Response	Server	Application	Session	ObjectContext Transaction Server

Request

L'oggetto **Request** gestisce l'input del codice. Si usa per accedere a tutte le informazioni passate attraverso una richiesta del browser client, come per esempio le variabili passate da un form tramite i metodi GET e POST, i Cookies o i ClientCertificate.

Response

L'oggetto Response gestisce gli output del codice; è usato dal server quando deve spedire al client le risposte relative a una sua richiesta. Per esempio invia informazioni al suo browser o reindirizza a un'altra pagina.

Server

L'oggetto **Server** fornisce l'accesso alle risorse del Server Web e una serie di funzionalità particolari.

Application

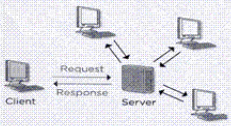
L'oggetto **Application** si usa per condividere informazioni tra tutti gli utenti di una data applicazione (per esempio il numero di utenti collegati a un sito in un certo momento).

Session

L'oggetto **Session** serve per memorizzare informazioni necessarie per una certa Sessione-Utente: le informazioni si perdono quando l'utente abbandona l'applicazione. L'oggetto Session rappresenta perciò la sessione corrente di ciascun utente che sta usando una particolare applicazione e permette di modificare le informazioni dell'utente stesso sul server per tutta la durata della connessione.

ObjectContext e Transaction Server

L'oggetto **ObjectContext** viene usato insieme al Microsoft **Transaction Server** per garantire che alcuni script siano gestiti in modo transazionale: se tutte le istruzioni dello script sono portate a termine lo script ha successo, altrimenti l'intero script fallisce. Le operazioni di modifica dei dati nei database devono essere di tipo transazionale per evitare di avere dati inconsistenti.



Nella versione ASP 3.0 è stato aggiunto un settimo oggetto, **ASPError**, che permette ai programmatori di leggere le proprietà che caratterizzano l'ultimo errore che si è verificato nello script correntemente in esecuzione.
In questa sezione vengono analizzati nel dettaglio gli oggetti **Request** e **Response** in quanto sono fondamentali per realizzare programmazione lato server, infatti il **primo** fornisce informazioni relative alle richieste inviate dal client all'applicazione Web mentre il **secondo** consente di inviare messaggi di risposta del server.

COMANDI ASP

Metodi dell'oggetto Request

Metodo form

È il metodo utilizzato per inviare dati al server tramite metodo `Post`.

Metodo Querystring

È il metodo utilizzato per inviare dati al server tramite metodo `Get`.
Analizzeremo entrambi i metodi nel prossimo paragrafo.

Metodi e proprietà dell'oggetto Response

Metodo Write

Il metodo **Write** è sicuramente il più utilizzato per inviare dati al client. I dati inviati vengono visualizzati come parte della pagina Web.

```
RESPONSE.WRITE "espressione"
```

Oppure

```
<%=espressione%>
```

Metodo End

Termina l'elaborazione dello script ASP e invia il contenuto del buffer al Browser.

Metodo Redirect

Utilizzato per passare dalla pagina in elaborazione a quella specificata dall'URL, la pagina può essere interna al sito o esterna.

```
Response.Redirect URL
```

Oppure

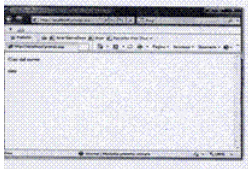
```
Response.redirect=URL
```

Proprietà Buffer

Il server, di default, invierebbe al client una pagina di risposta inserendo una riga per volta, man mano che interpreta le istruzioni dello script ASP.
Per inviare al client la pagina di risposta dopo averla costruita interamente in memoria e non riga per riga si deve inserire la seguente riga di codice all'inizio della pagina ASP:

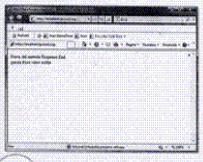
```
<% Response.Buffer=true%>
```

Esempi.
Il primo listato è una semplice prova del metodo **write**, mentre il secondo è una prova del metodo **end** dal quale si evidenzia che si chiude la sessione all'istruzione `response.end` e le frasi successive non vengono inviate al client.



```
<html>
  <body>
    Ciao dal server
    <br><br>
    <%
      response.write ("ciao")
    %>
  </body>
</html>
```

COMANDI ASP



```
<html>
<body>
Prova del metodo Response.End <br>
<%response.write("questa frase viene scritta")
    response.end
    response.write
  ("questa frase non viene scritta")%>
   neanche questa frase viene scritta!!
</body>
 </html>
```

COMPETENZA
SINTETIZZARE E SCHEMATIZZARE ATTRAVERSO L'USO DI MAPPE CONCETTUALI

2.3 L'INVIO E LA RICEZIONE DEI DATI CON I FORM

Come abbiamo detto nel primo paragrafo i form permettono di inviare dati al server tramite due metodi `Get` e `Post`, questo succede nel momento in cui l'utente clicca il pulsante di azione codificato con il tipo `Submit`; il server riceve i dati ed esegue il file indicato con la proprietà `Action`, ma per poter elaborare tale file è necessario che recuperi i dati dal form inviato, il server può fare questo tramite l'oggetto `Request`.

Invio e ricezione con metodo Get

Il metodo `Get` utilizza la stringa di interrogazione per spedire i dati, accodandoli alla barra dell'indirizzo URL e il server per poterli recuperare deve servirsi del seguente comando:

```
Request.querystring("nomedato")
```

Request: è l'oggetto ASP che permette al server di recuperare i dati dal form;
Querystring: è il metodo che indica al server dove recuperare i dati;
Nomedato: è il nome del campo del form nel quale i dati sono stati inseriti, corrisponde allo stesso nome utilizzato nella creazione del form stesso con l'attributo `Name`.

```
<html>
  <body>
    <form method="get" action="eseguiget.asp">
      IMMETTI IL TUO NOME <input type="tex"name="nome"> <br>
      IMMETTI IL TUO COGNOME <input type="text"
                              name="cognome"><br>
```

COMANDI ASP

```
    <input type="SUBMIT" VALUE="INVIA">
  </form>
 </body>
</html>
```

L'esempio prevede l'invio dei dati al server tramite metodo `get`; il server esegue il file **eseguiget.asp** e risponde con un messaggio di benvenuto.

Nella pagina Web di risposta vediamo che nella barra degli indirizzi accanto al nome del file eseguito compaiono tutti i dati del form:

- il carattere **?** separa il nome del file dal primo dato;
- successivamente appare la prima Collection **nome=Mario**;
- infine appare la seconda Collection **cognome=Rossi**;
- le due Collection sono separate dal carattere **&**.

Il codice della pagina è il seguente:

```
<%  option explicit
    dim nomeutente,cognutente
    nomeutente=request.querystring("nome")
    cognutente=request.querystring("cognome")
    response.write("BenVenuto " & nomeutente & " " & cognutente)
%>
```

Nel codice possiamo vedere che nel metodo `querystring` sono stati inseriti esattamente i nomi che compaiono nell'`input type` del form; nell'oggetto `response` vediamo inoltre una sintassi particolare: è stato usato il carattere `&` per separare le variabili visualizzate in modo da creare uno spazio tra loro.

Invio e ricezione con metodo Post

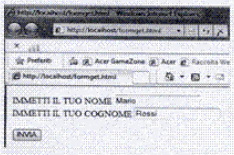
Il metodo `post` utilizza il form per l'invio dei dati, ma questi non vengono visualizzati sulla barra dell'URL. Il server recupera i dati tramite il seguente comando:

```
Request.form("nomedato")
```

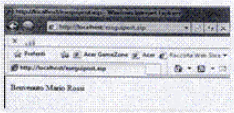
Request: è l'oggetto Asp che permette al server di recuperare i dati dal form;
Form: è il metodo che indica al server dove recuperare i dati;
Nomedato: è il nome del campo del form nel quale i dati sono stati inseriti, corrisponde allo stesso nome utilizzato nella creazione del form stesso con l'attributo `Name`.

COMANDI ASP

Vediamo ora lo stesso esempio di prima utilizzando il metodo `Post`:




```
<html>
 <body>
  <form method="post" action="eseguipost.asp">
   IMMETTI IL TUO NOME
   <input type="text" name="nome"><br>
   IMMETTI IL TUO COGNOME
   <input type="text" name="cognome"><br>
   <input type="SUBMIT" VALUE="INVIA">
  </form>
 </body>
</html>
```

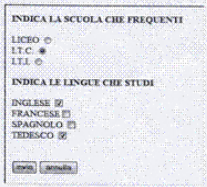


Il codice del form è analogo al precedente cambia solo il metodo (`post`) e il nome del file seguito (**eseguipost.asp**) in quanto sono due elaborazioni diverse. Il risultato si vede nella videata a lato.

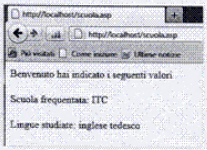
Nella barra dell'indirizzo URL non compare alcun dato ma solo il nome del file eseguito.
Il codice della pagina ASP è il seguente:

```
<%@ language=VBScript%>
<% option explicit
dim nomeutente,cognutente
nomeutente=request.form("nome")
cognutente=request.form("cognome")
response.write("Benvenuto " & nomeutente & " " & cognutente)
%>
```

Rispetto all'esempio precedente cambia solo il metodo del form.
L'esempio seguente utilizza un form con i pulsanti di opzione e usa il metodo `Post`:



Ipotizziamo che l'utente selezioni le caselle spuntate, il risultato ottenuto dal server è:



COMANDI ASP

Il codice delle pagine è il seguente:

- Codice del form di invio dati (**opzioni.html**):

```
<html>
  <head>
    <title> inserimento dati </title>
  </head>
  <body>
    <h4> INDICA LA SCUOLA CHE FREQUENTI</H4>
    <form method="post" action="scuola.asp">
    LICEO <input type="radio" name="scuola" value="LICEO"><br>
    I.T.C. <input type="radio" name="scuola" value="ITC"><br>
    I.T.I. <input type="radio" name="scuola" value="ITI"><br>
      <h4>INDICA LE LINGUE CHE STUDI</h4>
INGLESE<input type="checkbox" name="inglese" value="si"><br>
FRANCESE<input type="checkbox" name="francese" value="si">
<br>
SPAGNOLO <input type="checkbox" name="spagnolo" value="si">
<hR>
TEDESCO <input type="checkbox" name="tedesco" value="si">
      <br><br><br>
      <input type="submit" name="conferma" value="invia">
      <input type=reset name="cancella" value="annulla">
    </form>
  </body>
</html>
```

- Codice della pagina ASP (**scuola.asp**):

```
<%@ language=VBScript%>
<%  scuola=request.form("scuola")
      inglese=request.form("inglese")
      francese=request.form("francese")
      tedesco=request.form("tedesco")
      spagnolo=request.form("spagnolo")
      response.write ("Benvenuto hai indicato i seguenti valori")%>
<br><br>
<% response.write ("Scuola frequentata:" & scuola)%>
<br><br>
<% response.write ("Lingue studiate:")
       if inglese="si" then
            response.write ("inglese")
       end if
       if francese="si" then
            response.write ("francese")
       end if
       if spagnolo="si" then
            response.write ("spagnolo")
       end if
       if tedesco="si" then
            response.write ("tedesco")
       end if
    %>
```

Il codice risulta più complesso perché le scelta multipla richiede vari controlli per sapere cosa è stato selezionato. Notiamo che il codice ASP si interrompe più volte dove il tag `%` si chiude perché intercalato con codice HTML per andare a capo con `<br>`.

ASP.NET

La tecnologia ASP.NET e i database

Introduzione alle pagine dinamiche

Con il linguaggio *HTML* si possono creare delle pagine *Web statiche*.Se invece volessimo creare delle pagine *Web dinamiche*, pagine il cui codice html viene generato al volo da un server,allora dovremmo utilizzare il linguaggio *ASP* per la piattaforma NT-IIS e *PHP* per la piattaforma Linux - Apache. Le tipiche applicazioni che utilizzano le pagine Web dinamiche, sono le interrogazioni a data base remoti.

Accesso ai DataBase da pagine ASP

Per effettuare la comunicazione da una pagina ASP al database si utilizza il componente ADO che consente l'accesso a tutti i tipi di dati. ADO mette a disposizione diversi oggetti che possono essere utilizzati per la connessione al database e per la sua interrogazione: l'oggetto ***Connection, RecordSet, Error, Field, Command***.

1) l'oggetto ***Connection*** consente di stabilire la connessione con la sorgente di dati.

2) l'oggetto ***RecordSet*** consente di lavorare con i dati di una tabella, contenendo infatti un insieme di record della tabella stessa. Mediante questo oggetto possiamo leggere, modificare o aggiornare dati alla tabella.

3) l'oggetto ***Command*** combina l'oggetto RecordSet e l'oggetto Connection.

<u>Per estrarre dati da un database sono necessarie due fasi:</u>

- Stabilire la connessione al database;
- Effettuare la vera e propria interrogazione al database

Le pagine ASP.NET

Il codice ASP.NET è l'evoluzione del codice ASP classico ed è identificato dall'estensione **.aspx.**
È possibile eseguire pagine ASP e ASP.NET sullo stesso server Web senza interferenze. ASP.NET utilizza il linguaggio Visual Basic e la prima volta che il server riceve la richiesta di esecuzione di una pagina ASP.NET, questa viene compilata. Per utilizzare le pagine ASP.NET il computer deve avere installato il S.O. Windows 2000 o XP o 2003 e un Web Server oltre a **.NET Framework** (librerie software di Microsoft per le applicazioni Web).

Accesso ai DataBase da pagine ASP.NET

ADO.NET è la tecnologia di accesso ai dati che rappresenta un'evoluzione di ADO; tale tecnica è utilizzabile con batabase di piccole e grandi dimensioni.
In **ADO.NET** non è più disponibile l'oggetto RecordSet.

Gli oggetti principali di **ADO.NET** per costruire applicazioni Web, con accesso ai database in rete, sono:

1) ***Connection***, per stabilire la connessione al database
2) ***Command***, per eseguire i comandi di manipolazione o interrogazione al database
3) ***DataReader***, per ottenere i dati richiesti dal database.

I *DataReader* vengono poi associati ai ***controlli server*** di ADO.NET per visualizzare i dati in forma tabellare nelle pagine Web.

Tali controlli server sono:

DataGrid
Repeater
DataList.

Il metodo ***ExecuteReader*** applicato all'oggetto di tipo *Command* esegue il comando SQL, memorizzato in una stringa, e crea i dati all'interno dell'oggetto *DataReader*. Esso viene associato poi, attraverso il metodo ***DataBind***, al controllo definito nella pagina ASP.NET per la visualizzazione dei dati.

Per leggere e scrivere i dati occorre prima di tutto stabilire una connessione con il database attraverso un oggetto di tipo *Connection* che è diverso a seconda del database utilizzato: questo oggetto si chiama ***SqlConnection*** per i database SQLServer e **OleDbConnection** per i database Access.

Lo spazio dei nomi (***namespace***) per SqlServer è System.Data.SqlClient, per Access è System.Data.OleDb.

Per questo motivo le pagine ASP.NET che accedono ai database di Access contengono come prima riga la dichiarazione di importazione degli oggetti OleDB:

```
<%@ import Namespace="System.Data.OleDb" %>
```


Struttura di una pagina ASP.NET

```
<%@ Page Language="VB" %>
<script runat="server"> "parte eseguita  dal server"
Sub

end sub

</script>

<html>
<head>
</head>
<body>
<form runat="server">

</form>

</body>
</html>
```

1) la prima parte, compresa tra <script> e </script>, raggruppa le istruzioni per l'elaborazione sul server. Il codice è organizzato in sottoprogrammi (sub end sub).

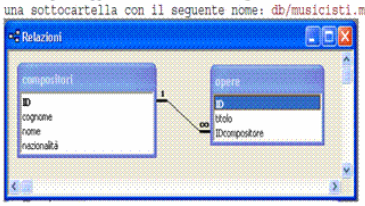
2) La seconda parte contiene gli elementi utilizzabili dal browser per presentare la pagina all'utente: tutti gli elementi sono raggruppati tra <form> e </form>.

Come gestire un data base on line con le pagine ASP.NET
ESEMPIO PRATICO

Dopo aver creato il database (con Access) e creato il progetto utilizzando Visual Basic, si può gestire il database anche **on line**.

In una cartella del sito che si sta amministrando si può creare un file **index.html** col ruolo di gestore delle varie opzioni da scegliere. Per rendere tale file più semplice possibile, potremo immaginarlo con solo dei pulsanti che attivano le relative **pagine ASP.NET** idonee a gestire le specifiche richieste.

Ad esempio supponiamo di avere il seguente database memorizzato in una sottocartella con il seguente nome: db/musicisti.mdb.



e di voler realizzare:
1) una query parametrica che trova tutti i compositori di una nazione
2) ed una che è in grado di inserire un nuovo compositore.

La codifica in SQL è la seguente:

```
1) SELECT compositori.cognome, compositori.nome,
   compositori.nazionalità FROM compositori WHERE
   (((compositori.cognome)=[inserisci la nazionalità]));

2) Insert Into nazionalità (cognome, nome, nazionalità) Values
   ([inserire cognome],[inserire nome],[inserire nazionalità]);
```

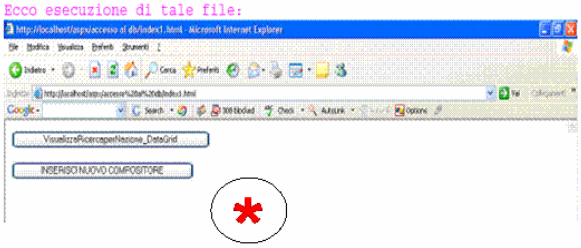
ASP.NET

Ecco quindi i relativi file per gestire ON LINE il BATABASE:

file index.html

```
<html>
<body>

    <form name="Pul_DataGrid" method="post"
    action="RicercaNazione_DataGrid.aspx">

        <input type="submit" name="Pul_visualizzaDG" value="Ricerca
        per Nazione_DataGrid">

   </form>


    <form name="Pul_Inserisci" method="post" action="inserisci.aspx">

     <input type="submit" name="Pulsante_inserisci" value="INSERISCI
NUOVO COMPOSITORE">

    </form>

</body>
</html>
```

Ecco esecuzione di tale file:



ASP.NET

File: RicercaNazione_DataGrid.aspx
Si utilizza il controllo DataGrid

Per realizzare la seguente query parametrica si utilizza il comando ***SELECT*** del linguaggio SQL per ottenere i dati da visualizzare.
Ogni volta che si deve utilizzare il comando SELECT si può utilizzare tale pagina ASP.NET modificando solo la stringa strSQL.

```
<%@ Page Language="VB" %>
<%@ import Namespace="System.Data.OleDb" %>
<script runat="server">

    ' Insert page code here
    sub Page_Load
    lbl1.text="Ricerca Musicisti per Nazione "
    end sub

    Sub submit_click(sender As Object, e As EventArgs)
    Dim dbconn as OleDbConnection
    Dim dbcomm as OleDbCommand
    Dim dbread as OleDbDataReader
    Dim strSQL as string
    dim nome as string

    nome = NomeNazione.Text

    dbconn = New OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;
data source=" & server.mappath("db/musicisti.mdb"))
    dbconn.Open()

    strSQL = "Select cognome, nome From compositori "
    strSQL = strSQL & "where nazionalità = '" & nome & "' order by
cognome"

    dbcomm = New OleDbCommand(strSQL , dbconn)

    dbread = dbcomm.ExecuteReader()

    dbconn.Close()    a=dbread
                      )
    End Sub

</script>
```

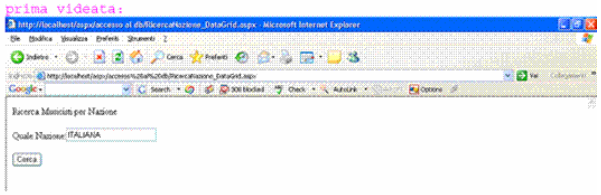

ASP.NET

```
<html>
<head>
</head>
<body>
    <form runat="server">
        <p>
            <asp:Label id="lbl1" runat="server">Ricerca</asp:Label>
        </p>
        <p>
            Quale Nazione:<asp:TextBox id="NomeNazione"
runat="server"></asp:TextBox>
        </p>
        <p>
            <asp:Button id="submit" onclick="submit_Click"
runat="server" Text="Cerca"></asp:Button>
        </p>
        <p>
        </p>
        <p>
            <asp:DataGrid id="musici" runat="server"


          headerstyle-font-name="Verdana"
          headerstyle-font-size="10pt"
          headerstyle-horizontalalign="center"
          headerstyle-font-bold="True"
          headerstyle-backcolor="#778899"
          headerstyle-forecolor="#ffffff"

          >
           </asp:DataGrid>

        </p>
        <!-- Insert content here -->
    </form>
</body>
</html>
```

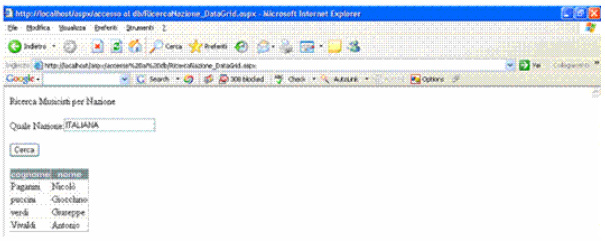

ASP.NET

Ecco esecuzione di tale file:

prima videata:



seconda videata:
dopo aver cliccato il pulsante "Cerca"



File: inserisci.aspx

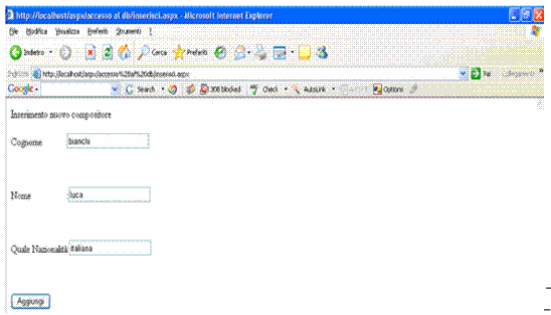
Con la seguente pagina ASP.NET possiamo usare i comandi di manipolazione ***Insert***, ***Update*** e ***Delete*** del linguaggio SQL.
L'esempio seguente mostra l'uso del comando Insert, ma la procedura presentata ha carattere generale in quanto, per gli altri comandi, basta sostituire la sola stringa contenente il comando SQL.
L'esecuzione del comando SQL è attivata attraverso il metodo ***ExecuteNonQuery***. Tale metodo inoltre restituisce un valore numerico che rappresenta il numero di righe (della tabella) interessate dal comandi SQL.
Si deve osservare che, per consentire l'esecuzione dei comandi di manipolazione sui database in rete, l'Amministratore del Web server deve impostare i permessi di scrittura, sul file contenente il database, per gli utenti che accedono tramite pagine Web.

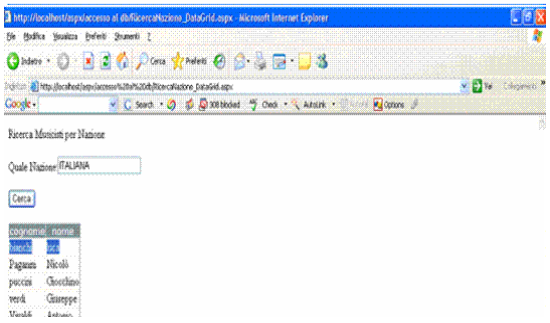
```
<%@ Page Language="VB" %>
<%@ import Namespace="System.Data.OleDb" %>
<script runat="server">

    sub Page_Load
    lbl1.text="Inserimento nuovo compositore "
    end sub

    Sub submit_click(sender As Object, e As EventArgs)

    Dim dbconn as OleDbConnection
    Dim dbcomm as OleDbCommand


    Dim strSQL as string
    dim numrec as integer


    dbconn = New OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;
data source=" & server.mappath("db/musicisti.mdb"))
    dbconn.Open()

    strSQL = "Insert Into compositori (cognome, nome, nazionalità) "
    strSQL = strSQL & "Values ("
    strSQL = strSQL & "'" & cognome.text & "', "
    strSQL = strSQL & "'" & nome.text & "', "
    strSQL = strSQL & "'" & nazionalità.text & "'"
    strSQL = strSQL & ")"

    dbcomm = New OleDbCommand(strSQL , dbconn)

    numrec= dbcomm.ExecuteNonQuery()
    lbl2.text = "Record aggiunti = " & numrec
    dbconn.Close()

    End Sub
```

ASP.NET

```
</script>
<html>
<head>
</head>
<body>
    <form runat="server">
        <p>
            <asp:Label id="lbl1" runat="server">Inserisci</asp:Label>
        </p>
        <p>
            Cognome          
 :<asp:TextBox id="cognome" runat="server"></asp:TextBox>
        </p>
        <p>
             </p>
        <p>
            Nome              
  :<asp:TextBox id="nome" runat="server"></asp:TextBox>
        </p>
        <p>
             </p>
        <p>


            Quale Nazionalità:<asp:TextBox id="nazionalità"
runat="server"></asp:TextBox>
        </p>
        <p>
             </p>
        <p>
            <asp:Button id="submit" onclick="submit_Click"
runat="server" Text="Aggiungi"></asp:Button>
        </p>
        <p>
            <asp:Label id="lbl2" runat="server"></asp:Label>
        </p>

    </form>
</body>
</html>
```

ASP.NET





R I E P I L O G A N D 0

Ogni pagina ASP inizia con l'indicazione del linguaggio di script attraverso il tag:

<%@LANGUAGE=Nome del linguaggio Script%>

Se l'utente utilizza VBScript il comando non è necessario in quanto è lo script utilizzato di default da ASP, altrimenti esso deve sempre comparire nella prima riga.

Tutti i comandi ASP devono essere delimitati dai tag **<%.....%>**

Per esempio una semplice pagina Asp può essere la seguente:

Il server invia al client una pagina web nella quale si visualizza la stringa **"Hello Word!!**; il file si chiama **ciaomondo.asp**, il nome è chiaramente visibile sulla barra dell'indirizzo ed è stata pubblicata sul server simulato **http:localhost"**

<%@LANGUAGE=VBSCRIPT%>

DICHIARAZIONE DEL LINGUAGGIO

<HTML>

<HEAD>

<TITLE> HELLO WORD! **</TITLE>**

</HEAD>

<BODY>

<%RESPONSE.WRITE "HELLO WORLD!"%> **CODICE ASP**

</BODY>

</HTML>